INTERAKTIVES PROBLEMLÖSEN IN MENSCH-MASCHINEN-KOMMUNIKATION

Dissertation
zur Erlangung des wirtschaftswissenschaftlichen Doktorgrades
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Georg-August-Universität zu Göttingen

vorgelegt
von
HELMUT ROTH
aus Wanfried

Göttingen 1976

Digitized by the Internet Archive
in 2026

https://archive.org/details/IA41534921_0034

INTERAKTIVES PROBLEMLÖSEN IN MENSCH-MASCHINEN-KOMMUNIKATION

Dissertation
zur Erlangung des wirtschaftswissenschaftlichen Doktorgrades
der Wirtschafts- und Sozialwissenschaftlichen Fakultät
der Georg-August-Universität zu Göttingen

vorgelegt
von
HELMUT ROTH
aus Wanfried

Göttingen 1976

UNIV.
LEIDEN
BIBL.

Berichterstatter: Prof. Dr. Hartmut Noltemeier

Mitberichterstatter: Prof. Dr. Jürgen Bloech

Tag der mündlichen Prüfung: 28.6.1976

Inhaltsverzeichnis

0. Zielsetzung und Überblick.....3

1. Konzeption und Verwendung interaktiver Problemlösungen.....6

1.1. Zum Begriff "interaktive Problemlösung".....6

1.2. Faktoren, die die Erstellung interaktiver Problemlösungen bedingen können.....8

1.2.1. Problemunabhängige Faktoren.....11

1.2.2. Problemabhängige Faktoren - dargestellt anhand betrieblicher Entscheidungsprobleme..14

1.2.2.1. Betriebliche Entscheidungsprobleme.....15

1.2.2.2. Interaktives Problemlösen bedingende Faktoren betrieblicher Entscheidungsprobleme.....21

1.3. Zur Verwendung interaktiver Entscheidungsproblemlösungen in Betrieben.....33

2. Beschreibung interaktiver Problemlösungen durch Modelle.....42

2.1. Interaktives Lösen einer Transportmitteldisposition.....42

2.1.1. Die Problemstellung und eine Problemlösung..44

2.1.2. Beschreibungsmerkmale dieser Problemlösung..50

2.2. Allgemeine Beschreibungsmerkmale interaktiver Problemlösungen.....53

2.3. Interaktive Prozeduren.....59

2.3.1. Formale Definition interaktiver Prozeduren..59

2.3.2. Interaktive Prozeduren und Turingprogramme..71

2.3.3. Darstellung eines Teils der interaktiven Transportmitteldisposition als interaktive Prozedur.....76

2.4. Zum zeitlichen Charakter interaktiver Problemlösungen.....79

3. Struktur und Eigenschaften interaktiver Prozeduren..83

3.1. Dialoge.....83

3.1.1. Formale Definition von Dialogen.....83

3.1.2. Dialoge innerhalb der interaktiven Transportmitteldisposition.....86

3.2. Zur Struktur interaktiver Prozeduren.....89

3.2.1. Grundkomponenten interaktiver Prozeduren....89

3.2.2. Grundkomponenten eines Ausschnitts der interaktiven Transportmitteldisposition.....98

3.3. Interaktive Prozeduren und nichtdeterministische Programme.....101

3.3.1. Auswählende Programme.....101

3.3.2. Ein Zusammenhang interaktiver Prozeduren mit auswählenden Programmen.....106

3.4. Zur Korrektheit interaktiver Problemlösungen...114

3.4.1. Korrektheit auswählender Programme.....116

3.4.2. Korrektheit interaktiver Problemlösungen...120

3.4.3. Korrektheit innerhalb der interaktiven Transportmitteldisposition.....124

3.5. Zur Komplexität interaktiver Problemlösungen...128

4. Schlußbemerkungen.....135

5. Literatur.....136

Notation.....149

Anhang 1 Die implementierte Transportmitteldisposition-Problemlösung.....150

Anhang 2 Dialogprotokoll eines Transportmitteldisposition-Lösungsprozesses.....162

0. Zielsetzung und Überblick

Lösen von Problemen in Kommunikation zwischen einem Menschen und einem Rechner hat zunehmend Bedeutung erlangt. Dies dokumentiert sich nicht zuletzt in der nahezu unüberschaubaren Fülle an Schilderungen interaktiver Problemlösungen für spezielle Anwendungen. Vergleichsweise gering nehmen sich daneben Überlegungen aus, die eine allgemeine Betrachtung interaktiver Problemlösungen, insbesondere ihrer Eigenschaften, zum Inhalt haben.

In der vorliegenden Arbeit wird nun zunächst eine Modellbildung für interaktive Problemlösungen vorgenommen. Das Ziel ist, eine formale Basis zu schaffen und von ihr aus Aussagen über Eigenschaften interaktiver Problemlösungen zu gewinnen. Unter dieser Zielsetzung gliedert sich die Arbeit wie folgt auf:

Nach einer Bestimmung des Begriffs "interaktive Problemlösung" (1.1.) werden Fähigkeiten eines Rechners und eines Menschen betrachtet, die in interaktiven Lösungsprozessen effizient einsetzbar sind (1.2.1.). Anhand ausgewählter betrieblicher Entscheidungsprobleme - sie werden in 1.2.2.1. in allgemeinem Rahmen erörtert - wird dann gezeigt, wie diese Fähigkeiten aufgrund problemspezifischer Eigenschaften die Konzeption interaktiver Entscheidungsproblemlösungen bestimmen (1.2.2.2.). Eine Diskussion betrieblicher Anwendungsmöglichkeiten interaktiver Problemlösungen, vor allem von Entscheidungsproblemen, schließt den ersten Teil der Arbeit ab.

Auf den konzeptionellen Erörterungen aus 1. aufbauend, dient der zweite Teil der Arbeit der Erstellung eines neuartigen Modells für interaktive Problemlösungen. Dazu wird zunächst eine interaktive Problemlösung für ein betriebliches Transportmittel-Dispositions-Problem erstellt (2.1.). Ihre Beschreibungsmerkmale (2.1.2.) sowie die in der Literatur beschriebener interaktiver Problemlösungen bilden die Grundlage für einen Katalog allgemeiner Beschreibungsmerkmale (2.2.). Er ist die Basis für ein Mensch und Rechner integrierendes formal abstraktes Modell für eine interaktive Problemlösung, "interaktive Prozedur" (Definition 2.3.1.1.) genannt; eine Formalisierung interaktiver Lösungsprozesse wird in Definition 2.3.1.3. vorgenommen. Nach einer Diskussion des Modells schließen sich in 2.3.2. erste Aussagen über seine Struktur an. In 2.3.3. wird ein Ausschnitt aus der interaktiven Transportmitteldisposition als interaktive Prozedur dargestellt. Eine Betrachtung zur Einbeziehung des Faktors Zeit in ein Modell zur Beschreibung interaktiver Problemlösungen (2.4.) beendet den zweiten Teil der Arbeit.

In Teil 3. schließen sich Aussagen über Eigenschaften interaktiver Problemlösungen auf der Basis des Modells "interaktive Prozedur" an; diese werden weitgehend an der interaktiven Transportmitteldisposition verdeutlicht. Neben formal exakten Begriffsbestimmungen (Dialoge, 3.1.) erlaubt eine Untersuchung der Struktur interaktiver Prozeduren

(3.2.) Aussagen über Implementierung und Dokumentation interaktiver Problemlösungen. Nun beeinhaltet aber die Anwendung interaktiver Problemlösungen auch über deren Struktur hinausgehende Probleme, wie die "Richtigkeit" und den "Aufwand" führbarer Lösungsprozesse; diese Probleme werden in den Teilen 3.4. und 3.5. der Arbeit angesprochen. Dazu wird zunächst in 3.3. ein Zusammenhang zwischen interaktiven Prozeduren und sogenannten auswählenden Programmen, besonderen nichtdeterministischen Programmen, hergestellt (Satz 3.3.2.1.); er ermöglicht Korrektheitsbetrachtungen für interaktive Problemlösungen auf der Basis der Korrektheit nichtdeterministischer Programme (3.4.2.). Der Aufwand von Lösungsprozessen in interaktiven Lösungsprozessen wird dann in 3.5. betrachtet; interaktive Prozeduren erweisen sich dabei als ein Mittel, mit dem die Komplexität eines interaktiven Lösungsprozesses relativ zu der von einem Menschen in ihn eingebrachten "intellektuellen Leistung" gemessen werden kann.

Eine Dokumentation der interaktiven Transportmitteldisposition sowie von Dialogen in ihr durchgeführter Lösungsprozesse (Anhang 1 und 2) schließen neben ausgewählter Literatur die Arbeit ab.

1. Konzeption und Verwendung interaktiver Problemlösungen

1.1. Zum Begriff "interaktive Problemlösung"

Ein Problem kann man - mengentheoretisch gesehen - nach (NEWELL-SIMON,1972) in folgender Weise auffassen[1]: In einer gegebenen Menge U von "möglichen Lösungen" sollen "Lösungen", d.h. Elemente einer Teilmenge G von U, die gewisse spezifizierte Eigenschaften erfüllen, bestimmt werden[2]. Das "Lösen" eines derart formulierten Problems besteht in der Bestimmung von Elementen aus G; ein solcher Bestimmungsprozeß ist ein Problemlösungsprozeß. Eine Problemlösung[3] ist die Angabe einer Verfahrensweise - etwa eines Algorithmus' - für die Durchführung von auf das Finden[4] von Lösungen hin zielgerichteten Lösungsprozessen.

Interaktives Problemlösen soll damit folgende Bedeutung[5] haben: "Lösen von Problemen in auf das Bestimmen von Lösungen

---

[1] Vgl. (NEWELL-SIMON,1972), p.73ff; auf Definitionen des Problembegriffs und die damit verbundene Problematik - siehe dazu etwa (NEWELL-SIMON,1972), p.72,73 - braucht hier nicht weiter eingegangen zu werden.

[2] Zur Problematik der Präzisierung der Begriffe "gegeben" und "bestimmen" siehe (NEWELL-SIMON,1972), p.74ff.

[3] Es ist zwischen Lösung eines Problems und Problemlösung zu unterscheiden; eine Lösung ist der (erfolgreiche) Abschluß eines Lösungsprozesses, der im Rahmen einer Problemlösung durchführbar ist.

[4] Vgl. dazu die Darstellung von Problemlösungsprozessen als Suchprozesse in geeigneten Zustandsräumen in (NEWELL-SIMON,1972), p.75ff, (NILSSON,1971), insbesondere in Kapitel 3 und 4.

[5] Vgl. (MEADOW,1970), p.4.

hin zielgerichteten Kommunikationsprozessen[1] zwischen einem Menschen und einem sich deterministisch verhaltenden (programmierten) digitalen Rechner"; beide werden im folgenden stets kurz als Benutzer[2] bzw. Rechner bezeichnet. Eine interaktive Problemlösung besteht dann aus der Angabe einer Verfahrensweise für Rechner und Benutzer zum Bestimmen von Lösungen in Kommunikationsprozessen zwischen ihnen. Für die Beschreibung derartiger Verfahrensweisen wird im Verlauf der Arbeit ein formales Modell entwickelt und diskutiert.

Bedingt durch die im Gang befindliche Entwicklung des Gebietes "interaktives Problemlösen" hat sich bislang keine einheitliche Terminologie herausgebildet; weitere, mehr oder weniger synonyme Begriffe sind: Problemlösen im Dialog, Mensch-Maschinen-Kommunikation, bzw. "conversational problem-solving" und "man-computer problem-solving"[3].

---

[1] Für eine Diskussion der Begriffe "Kommunikation" und "Kommunikationsprozeß" vgl. (DIRLEWANGER, et al.,1972), Teil 1.

[2] Ein Benutzer kann dabei auch eine einheitlich handelnde Gruppe von Personen sein.

[3] Auch unter mit den Begriffen "on-line" und "Echtzeit" bzw. "real-time" charakterisiertem Problemlösen wird bisweilen implizit interaktives Problemlösen verstanden; die damit verbundene Begriffsvermischung ist jedoch abzulehnen.

1.2. Gründe, die die Erstellung interaktiver Problemlösungen bedingen können

In einer interaktiven Problemlösung können zwei Grenzfälle wegen fehlender Kommunikation auftreten: ein Rechner führt eine etwa durch ein auszuführendes Programm gegebene Problemlösung selbsttätig mit gegebenen Daten durch, oder ein Benutzer löst sein Problem allein ohne Kommunikation mit einem Rechner. Der zweite Grenzfall wird in dieser Arbeit nicht mitbetrachtet, da er eine reine Behandlung menschlichen Problemlösungsverhaltens[1] bedeuten würde, was nicht Ziel der folgenden Ausführungen ist.

Damit bleiben interaktive Problemlösungen übrig, die mindestens einmal in einem Lösungsprozeß die Mitwirkung eines Rechners vorsehen und somit als "rechnerunterstützt"[2] bezeichnet werden können. Dies bedingt aufgrund der hier betrachteten Rechner, vgl.1.1., daß zumindest für eine Lösungskomponente eine genaue Modell- und Methodenvorstellung auf einem Rechner implementiert ist[3]. In einem interaktiven Lösungsprozeß können dann neben den implementierten Modellen, Methoden und sonstigen vorhandenen Daten von einem Benutzer weitere Modelle,

---

[1] Siehe dazu etwa (NEWELL-SIMON,1972).

[2] Synonym: computerunterstützt, computergestützt.

[3] Zum Begriff Modell und der Problematik einer Modellbildung siehe (KIRSCH,1971), p.32ff., (DINKELBACH,1973), (WINAND, 1975); zum Begriff Methode siehe (NEWELL-SIMON,1972), p.88.

Methoden sowie Daten eingebracht werden, um den Lösungsprozeß gemäß seiner Vorstellungen[1] zu beeinflussen; derartige Aktivitäten eines Benutzers können sich dabei auch erst im Verlauf des Lösungsprozesses herausbilden. Die implementierten und die in einem interaktiven Lösungsprozeß von einem Benutzer eingebrachten Modelle und Methoden sowie weitere Daten können dabei Veränderungen unterliegen[2]; allerdings sollte die Konzeption einer interaktiven Problemlösung im Interesse "rationalen" Problemlösens so sein, daß in den in ihr führbaren Lösungsprozessen keine Situationen auftreten können, in denen für Benutzer oder Rechner Ungewißheit über ihre zielgerichteten - vgl.1.1. - Fortsetzungsmöglichkeiten besteht.

Die Zusammenhänge zwischen im Rechner implementierten Modellen, Methoden und weiteren Daten einerseits und denen eines Benutzers bei der Durchführung eines Lösungsprozesses andererseits sollen die folgenden Beispielüberlegungen verdeutlichen. Betrachtet man eine APL-Maschine - damit sei eine auf einem realen Rechner implementierte Version der Dialogsprache APL[3]

---

[1] Vgl. etwa (CARBONELL,1969), p.17: "The HO (Human Operator, d.Verf.) usually approaches the console with a plan of work, broad and sketchy or detailed and specific".

[2] Vgl. dazu insbesondere Verfahren, die menschliche Denkprozesse simulieren sollen, wie etwa "heuristische" Programme (KIRSCH,1971),p.153ff., (KLEIN,1971), p.38ff., "lernende" Programme (KIRSCH,1971), p.74,75 innerhalb "adaptiver" Verfahren (GRIESE,1972), p.14ff.

[3] Zu APL siehe (GILMAN-ALLEN,1973), (UNIVAC,1973).

bezeichnet - so ist sie als eine interaktive Lösung des Problems anzusehen, einem Benutzer im Rahmen der in ihr realisierten Modelle und Methoden Lösungen seiner eigenen Probleme zu ermöglichen. Dies kann dann etwa durch eine Benutzung der Komponenten der APL-Maschine, die einen Tischrechnermodus ermöglichen geschehen, indem ein Benutzer als Lösung eines Problems eine Folge von arithmetischen Operationen in einem Kommunikationsprozeß mit der APL-Maschine durchführt.

Es ist aber auch möglich, daß ein Benutzer mit den Komponenten der APL-Maschine, die einen Funktionsdefinitionsmodus ermöglichen, einen Kommunikationsprozeß zur Erstellung von (interaktiver) Software zur Realisierung einer Lösung eines Teilproblems im Rahmen seiner Modell- und Methodenvorstellungen innerhalb seines Problems führt. In diesem Fall erstellt der Benutzer eine neue "erweiterte" APL-Maschine, indem er zur ursprünglich gegebenen APL-Maschine seine Modell- und Methodenvorstellungen implementiert und so eine (interaktive) Lösung seines eigenen Problems konstruiert[1)]. Dieser Vorgang ist wiederholbar.

Die Konzeption einer interaktiven Problemlösung - wie etwa die APL-Maschine - zu einem vorgegebenen Problem geschieht nun (zumindest sollte sie so erfolgen) unter dem einem Problem übergeordneten Gesichtspunkt, Eigenschaften eines

---

1) Für Möglichkeiten einer Anwendung dieser allgemeinen Erörterungen siehe (DIERS-HABERLANDT,1973), insbesondere p.513 ff., (SCHOBER,1974), (GLATTHAAR,1974) - Einsatz von APL im rechnerunterstützten Unterricht, u.a. bei der Bildung von Modellen.

Menschen und eines Rechners so auszunutzen, daß eine bezüglich vorgegebener Kriterien möglichst effiziente Problemlösung erreicht wird.

1.2.1. Problemunabhängige Gründe

Es wird nun die Situation betrachtet, daß ein vorgegebenes Problem unter Einsatz eines Rechners gelöst werden soll. Dabei wird man sinnvollerweise versuchen, die besonderen Stärken eines Rechners und seines Benutzers im Sinne einer "man-computer-symbiosis"[1] zu kombinieren[2].

Die Stärken eines Rechners liegen in[3]

- seiner Sicherheit und Schnelligkeit bei der Durchführung auch komplexer vorgegebener Operationen,
- der Speicherung und Verarbeitung großer Datenmengen.

Die Stärken eines Benutzers liegen in[4]

- seiner Kreativität,
- seiner Fähigkeit, in Kontexten zu denken, was das Erkennen von Strukturen - auch versteckten - und das Auffinden von Informationen durch Assoziationen erlaubt[5],

---

[1] Vgl. (LICKLIDER,1960).

[2] Vgl. (MERTENS-KRESS,1970), p.2, (KIESER,1973), p.37, (MARTIN,1973), p.7, u.a..

[3] Vgl. (MERTENS-KRESS,1970), p.3ff., (KIESER,1973), p.106, (MARTIN,1973), p.7.

[4] Vgl. (MERTENS-KRESS,1970), p.4ff., (HORMAN,1971), p.167, (MILLER,1969), p.224,225, (SZYPERSKI-WINAND,1972), p.5, (KIESER,1973), Teil2.122, (MARTIN,1973), p.7.

[5] Für eine Diskussion des Begriffes Information, insbesondere auch unter betriebswirtschaftlichen Gesichtspunkten siehe (WACKER,1971), p.37ff.

- seinem Gefühl für Informationen anhaftende Ungewißheiten[1], die er in Wahrscheinlichkeiten umsetzen kann.

Man kann also sagen:

a. Ein Rechner ist in einer Problemlösung effizient einsetzbar, wenn Operationen auf vorgegebene Weise mit hoher Geschwindigkeit und Sicherheit, auch auf großen Datenmengen durchzuführen sind.

b. Ein Benutzer kann effizient in Problemlösungen einbezogen werden, wenn seine Stärken in nichtprogrammierbaren oder nichtprogrammierten Schritten wirksam werden können.

Nicht programmierbar bedeute dabei, daß - ggf. subjektiv gesehen - für eine Problemstellung keine Modelle und/oder Methoden und/oder Daten existieren, die eine Problemlösung nur für einen Rechner möglich machen würde.

a. und b. sind Kriterien für eine möglichst effiziente Aufteilung von Komponenten einer Problemlösung zwischen einem Rechner und seinem Benutzer[2]. Gleichzeitig sind allerdings bei der Konzeption einer interaktiven Problemlösung auch die Schwächen eines Menschen zu beachten; dies sind vor allem[3]:

---

[1] Zum Begriff der ungewissen Information siehe (WACKER,1971), p.53,54, (BAMBERG-COENENBERG,1974), p.17ff.

[2] Vgl. (MARTIN,1973), p.7: "This difference in "thinking" talent - the computer being good for ultrafast sequentiell logic and the human being capable of slow but highly parallel and associative thinking - is the basis for cooperation between man and machine"; vgl auch (HORMAN,1971).

[3] Vgl. (KIESER,1973), p.105ff.; für eine ausführliche kritische Betrachtung der Fähigkeiten eines Benutzers siehe (MARTIN,1973), sec.IV, insbesondere chap.17,19 und 20.

- Schwächen bei der Speicherung von (vielen) Details,
- Einbeziehung emotionaler Aspekte,
- geringe Rechenfähigkeiten;

bezüglich eines Rechners ist zu sehen, daß er bislang in nur geringem Maß "selbstständig" menschliche Denkprozesse simulieren[1] kann.

Eine weitere Schwierigkeit für eine Aufteilung von Lösungskomponenten zwischen einem Rechner und seinem Benutzer ergibt sich daraus, daß beide Kriterien a. und b. offenbar nicht scharf voneinander abgrenzbar sind. Insbesondere sind prinzipiell alle Lösungskomponenten, die ein Rechner durchführen kann, auch von einem Menschen durchführbar - wenn man von den ihm gegebenen Beschränkungen in Bezug auf Operationssicherheit, Geschwindigkeit und Handhabung großer Datenmengen absieht -, während dies umgekehrt nicht der Fall ist. Demzufolge ist eine scharfe Abgrenzung zwischen programmierbaren und nichtprogrammierbaren Lösungskomponenten, die zudem noch subjektiv beeinflußt werden kann[2], nicht möglich. Daraus können zwei entgegengesetzte Fehlverhalten resultieren: Einerseits kann versucht werden, einem Rechner möglichst

---

[1] Zu in diese Richtung weisenden Lösungsansätzen vgl. Fußnote 2 auf p.9.

[2] Objektiv nicht programmierbar ist etwa das nicht entscheidbare Halteproblem für Turingmaschinen, vgl. (HOPCROFT-ULLMAN,1969), Teil 7.3.; subjektiv nicht programmierbar ist etwa ein betriebliches Entscheidungsproblem, für das der Entscheidungsträger eine Problemlösung auf der Basis von implementierbaren Modellen und Methoden ablehnt, vgl. dazu auch Teil 1.2.2.2.

viele Aufgaben zu übertragen, wozu J. Martin warnend bemerkt[1]: "However it is ... important, that systems designers and those managers and other persons who think about computer usage do not try to make the computer compete with man in areas in which man is superior". Andererseits besteht die Gefahr, "daß durch zu starke Konzentration auf die MMK (Mensch-Maschinen-Kommunikation, d.Verf.) Situationen übersehen werden, in denen die Vollautomation die beste Lösung ist"[2].

1.2.2. Problemabhängige Faktoren - dargestellt anhand betrieblicher Entscheidungsprobleme

In Teil 1.2.1. sind Schwerpunkte eines möglichst effizienten Einsatzes von einem Rechner und Benutzer in einer interaktiven Problemlösung problemunabhängig diskutiert worden. Anhand ausgewählter betrieblicher Entscheidungsprobleme[3] wird nun betrachtet, wie diese Schwerpunkte aufgrund problemspezifischer Eigenschaften, sowie Einflüssen aus der betrieblichen

---

[1] Vgl. (MARTIN,1973), p.7.

[2] Vgl. (MERTENS-KRESS,1970), p.8.

[3] Vgl. (MILLER,1969), der Entscheidungsfindung allgemein als "archetype in man-computer problem solving" ansieht.

Umwelt[1] die Konzeption interaktiver Problemlösungen bestimmen.

1.2.2.1. Betriebliche Entscheidungsprobleme

Entscheidungsprobleme und ihre Lösung bestimmen wesentlich Zustand und Geschehen eines Betriebes[2]. Sie entstehen dadurch, daß das Verfolgen von Zielen - nach Heinen[3] "gibt ein Ziel allgemein einen zukünftigen Stand der Unternehmung[4] wieder, der als erstrebenswert angesehen wird" - Probleme ergibt. Lösen eines Entscheidungsproblems besteht dann - vgl. die allgemeinen Betrachtungen in 1.1. - im Suchen von Lösungen des Problems, den Entscheidungen[5], in einer Menge

---

[1] Eine problemspezifische Eigenschaft eines betrieblichen Entscheidungsproblems ist etwa die Angabe, in welcher Funktion eines Betriebes es anfällt; ein Einfluß aus der betrieblichen Umwelt ist etwa eine verfügbare Rechnerkonfiguration. Beide zusammen bilden Beschränkungen, denen mögliche Entscheidungen genügen müssen; vgl. dazu allgemein (KIRSCH, 1971), p.148ff.

[2] Vgl. dazu (WÖHE,1973/74), p.58: "Wirtschaftliches Handeln besteht seit jeher im Treffen von Entscheidungen", (HEINEN, 1966), p.18: "Wirtschaftliches Handeln erfordert ständiges Wählen zwischen verschiedenen Möglichkeiten".

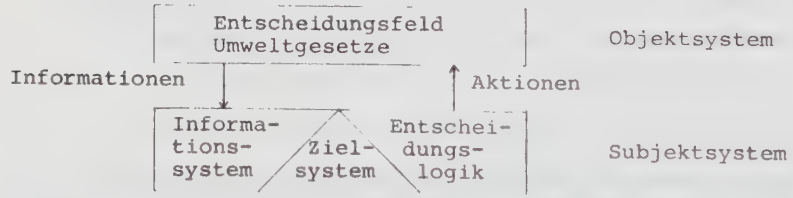
[3] Vgl. (HEINEN,1966), p.45.

[4] Hier braucht auf Unterschiede zwischen den Begriffen Unternehmung (im Zitat) und Betrieb (in der Arbeit stets benutzt) nicht eingegangen zu werden.

[5] Für eine ausführliche in die Literatur eingebettete Diskussion des Begriffs Entscheidung anhand ihn prägender Merkmale siehe (KAHLE,1973), p.17ff.

möglicher Entscheidungen[1]; ein dabei geführter Lösungsprozeß ist ein Entscheidungsprozeß[2]. Mißt man nun die Güte möglicher Entscheidungen in Bezug auf die verfolgten Ziele mit einem "Zielerfüllungsgrad", so lassen sich (betriebliche) Entscheidungsprobleme - operationell gesehen - in die folgende Frage kleiden: "Wie ist in einer konkreten Situation vorzugehen, so daß ein größtmöglicher Zielerfüllungsgrad erreicht wird?"[3]

Einen Entscheidungsprozeß kann man nach (BAMBERG-COENENBERG, 1974), (GÄFGEN,1968) in Grundzügen als die im folgenden beschriebene Wechselwirkung zwischen einem Objekt- und einem Subjektsystem angeben:



Figur 1.2.2.1.1.[4]

---

[1] Bei der Betrachtung eines "geschlossenen" Modells des Entscheidungsverhaltens - vgl. (KIRSCH,1970), p.25ff. - wird die Menge möglicher Lösungen einschließlich ihrer Konsequenzen als bekannt vorausgesetzt, nicht aber bei "offenen" Modellen des Entscheidungsverhaltens - vgl. (KIRSCH,1970), p.61ff. -, die das Empfinden von Problemen und das Suchen nach Informationen zu ihrer Lösung einschließen.

[2] Zu Problemlösungs- und Entscheidungsprozessen vgl. (KIRSCH, 1970), p.70ff.; da hier von einem gegebenen Entscheidungsproblem ausgegangen wird, braucht die Phase "Identifizierung des Problems" ((KIRSCH,1970), p.73) bei dem Erstellen einer Entscheidungsproblemlösung hier nicht betrachtet werden.

[3] Vgl. (BAMBERG-COENENBERG,1974), p.11.

[4] Vgl. (BAMBERG-COENENBERG,1974), p.1; für eine ähnliche Darstellung siehe (GÄFGEN,1968), p.96, (ZANGEMEISTER,1973), p.105.

Unter dem Objektsystem sind dabei "die unter dem Begriff Entscheidungsfeld zusammenzufassenden Aktionsmöglichkeiten, sowie die Bedingungskonstellation, die den Erfolg der Aktion beeinflussen, ferner Gesetzmäßigkeiten, nach denen sich das Entscheidungsfeld aus sich selbst heraus und unter dem Einfluß der Aktionsmöglichkeiten verändern kann"[1] zu verstehen. Das Subjektsystem enthält "die eigentlichen Entscheidungsdeterminanten"[1]; es enthält ein Informationssystem, das zur Aufstellung von Handlungsalternativen und zur Ermittlung ihrer Konsequenzen dient[1]; es werden damit "faktische Entscheidungsprämissen geschaffen"[1]. Weiterhin enthält das Subjektsystem ein Zielsystem, das Prämissen für die Bewertung der Konsequenzen von Handlungsalternativen liefert, sowie Kriterien für die Richtung der Informationsgewinnung im Informationssystem[1]. Die letzte Komponente des Subjektsystems ist dann die Entscheidungslogik, worunter ein mit der Auswahl einer Handlungsalternative endender "Prozeß der entscheidungslogischen Informationsverarbeitung"[1] zu verstehen ist.

Ein Entscheidungsprozeß setzt sich somit aus einem Informationsprozeß zur Erstellung eines Informationssystems, einem Zielbildungsprozeß zur Erstellung eines Zielsystems und einem Auswahlprozeß zur Auswahl von Handlungsalternativen zusammen, die sich gegenseitig beeinflussen.

---

[1] Vgl. (BAMBERG-COENENBERG,1974), p.1ff., (GÄFGEN,1968), p.95 ff.; vgl. auch (KIRSCH,1970), insbesondere die Betrachtungen zu "offenen" Modellen der Entscheidungsfindung, p.61ff.

Berücksichtigt man nun, daß ein betrieblicher Entscheidungsprozeß einen informationsverarbeitenden Prozeß im Informationssystem eines Betriebes[1] darstellt, so lassen sich seine Komponenten (schwerpunktmäßig) auch wie folgt beschreiben[2]:

Informationsprozeß: Gewinnung von (allen[3]) Handlungsalternativen, die im Rahmen der aus den verfolgten Zielen abgeleiteten Zielgrößen[4] relevant sind und Ermittlung ihrer Konsequenzen;

Zielbildungsprozeß: Generierung von Prämissen zur Bewertung von Handlungsalternativen und Bewertung der im Informationsprozeß bestimmten Handlungskonsequenzen; ggf. erfolgt ein Anstoß, den Informationsprozeß fortzusetzen oder neu zu beginnen;

---

1) Zum Informationssystem eines Betriebes siehe (WACKER,1971), insbesondere p.49ff.; (man beachte die unterschiedliche Verwendung des Begriffs Informationssystem hier und in Figur 1.2.2.1.).

2) Vgl. die (zweistufige) Unterteilung eines Entscheidungsprozesses in (KLEIN,1971).

3) Die "klassische" Entscheidungstheorie geht davon aus, daß alle Handlungsalternativen einem Entscheidungsträger bekannt sind, vgl. (KIRSCH,1970), p.27, (KLEIN,1971), p.64. Der betrieblichen Realität besser angenähert sind jedoch die schon erwähnten "offenen" Modelle des Entscheidens, die die Informationsgewinnung einschließen; diese orientiert sich an dem Informationsbedarf - dieser ist als Antwort auf die Frage: "Über wieviel Informationen muß ein Entscheidungsträger mindestens verfügen, wenn der Entscheidungsprozeß so rational bzw. so gut wie möglich sein soll?", vgl. (WACKER,1971), p.159,160, zu sehen - eines Entscheidungsträgers; seine Deckung wird bestimmt durch das Informationsverhalten des Entscheidungsträgers; dazu und zu empirisch abgesicherten Hypothesen über das Informationsverhalten von Entscheidungsträgern siehe (WITTE, 1972).

4) Zum Begriff Zielgröße vgl. (BAMBERG-COENENBERG,1974), p.26.

Auswahlprozeß: Auswahl einer Handlung aus den bestimmten Handlungsalternativen aufgrund ihrer bewerteten Konsequenzen; ggf. erfolgt eine Rückkopplung auf den Zielbildungs- und Informationsprozeß.

Für ein betriebliches Entscheidungsproblem treffen in der Regel die folgenden Merkmale zu:

1. Als erstes Merkmal stellt sich heraus, "daß die im praktischen Geschehen tatsächlich bedeutsame Information die relativ unvollkommene Information ist; und in diesem Rahmen vielleicht sogar die relativ ungewisse Information, allenfalls die relativ unsichere Information"[1]. Als Gründe dafür sollen hier nur die "Begrenzung der Informationsgewinnungs- und Informationsverarbeitungskapazität"[2] eines Menschen sowie aus der betrieblichen Umwelt eines Entscheidungsproblems herrührende Zwänge (Informationsgewinnungskosten, u.ä.) zur Beschränkung der Informationsgewinnung[3] genannt werden.

---

[1] Vgl. (WACKER,1971), p.54; der Begriff unvollkommene Information wird von Wacker als Oberbegriff für unsichere - im Sinne von mit meßbarem Risiko behaftete - und ungewisse - im Sinne von nicht angebbarem Risiko behaftete - Information benutzt. Für die verschiedenen Informationseigenschaften siehe auch (GÄFGEN,1968), p.129ff., (KIRSCH,1970), p.29, (BAMBERG-COENENBERG,1974), p.17ff.

[2] Vgl. (BAMBERG-COENENBERG,1974), p.6; vgl auch (HEINEN, 1966), p.159.

[3] Damit ist in der Regel nicht mehr die Bestimmung aller Handlungsalternativen und deren Konsequenzen möglich; siehe dazu auch Fußnote 3 auf p.18.

2. Das erste Merkmal ist der wesentliche Grund für das zweite Merkmal, daß jede Entscheidung auf relativ unvollkommener Information beruht[1], was "eine spezifisch ökonomische Problematik"[2] ist. Insbesondere implizieren die Beschränkungen in der Informationsgewinnungs- und Informationsverarbeitungskapazität, daß "neben dem eigentlichen Auswahlprozeß (stehen) damit in der Verhaltenstheorie der Unternehmung das Zustandekommen der faktischen und wertenden Entscheidungsprämissen, nämlich der Prozeß der Informationsgewinnung , sowie der Zielbildungsprozeß, im Mittelpunkt der Betrachtung"[3] stehen.

Beide Merkmale legen nahe, daß im Prozeß der Informationsgewinnung sowie im Zielbildungsprozeß in besonderem Maße Schwerpunkte interaktiver Arbeitsweise in einem rechnerunterstützt gelösten Entscheidungsproblem liegen werden. Anhand einiger ausgewählter Problemlösungen betrieblicher Entscheidungsprobleme wird dies nun verdeutlicht.

---

[1] Vgl. (WACKER,1971), p.54.

[2] Vgl. (WACKER,1971), p.56.

[3] Vgl. (BAMBERG-COENENBERG,1974), p.6, (HEINEN,1966), p.55ff.

1.2.2.2. Interaktives Problemlösen bedingende Faktoren betrieblicher Entscheidungsprobleme[1)]

Im Informationsprozeß eines Entscheidungsprozesses treten insbesondere Datenbeschaffungs- und Datenbewertungsprobleme[2)] auf, die bereits vom Informationsprozeß her ein vollständiges Übertragen eines Entscheidungsprozesses an einen Rechner aufgrund unvollständiger oder fehlender Daten verhindern können:

In (LODISH,1971) wird die interaktive Planung[3)] von Verkaufsbesuchen von Vertretern mit Hilfe des Programmes CALLPLAN beschrieben. Mit der Unterstützung von CALLPLAN ist es einem Vertreter möglich, für ein von ihm vertretenes Gut innerhalb eines Planungszeitraumes eine bezüglich maximaler Verkäufe optimale Besuchspolitik bei seinen Kunden und weiteren potentiellen Käufern in der Zielgruppe des Gutes zu planen. Die Problemlösung gliedert sich in zwei Teile:

Im ersten werden Daten - neben organisatorischen Angaben über Verkaufsbezirke vor allem Schätzwerte über erzielbare Verkäufe

---

[1)] Die im folgenden betrachteten interaktiven Problemlösungen betrieblicher Entscheidungsprobleme setzen sämtlich voraus, daß ein Benutzer während eines darin geführten Lösungsprozesses mit einem Rechner "unmittelbar" verbunden ist (dies ist im wesentlichen die Bedeutung des Begriffes "on-line"), und dieser in "vernachlässigbar kurzer Zeit" reagiert, vgl. etwa (MERTENS-KRESS,1970), p.1,2. Diese Voraussetzung wird hier in 1.2.2.2. berücksichtigt, da sie "üblicher" Betrachtungsweise interaktiven Problemlösens entspricht; generell wird in dieser Arbeit jedoch eine allgemeinere Auffassung des Begriffs interaktives Problemlösen vertreten, vgl.1.1.

[2)] Die Begriffe Daten und Informationen werden synonym behandelt; vgl. dazu (WACKER,1971), p.39.

[3)] Es soll hier und im folgenden nicht zwischen Planen und Entscheiden unterschieden werden, obwohl in der Literatur Planen bisweilen auch Durchführung und Kontrolle einer Entscheidung einschließt.

in Abhängigkeit von den Verkaufsbesuchen - erarbeitet. Im zweiten werden auf der Grundlage der erarbeiteten Daten nichtlineare Optimierungsprobleme formuliert und mit aus der nichtlinearen Optimierung entnommenen Methoden gelöst.

Die Erarbeitung der Daten geschieht dadurch, daß der Entscheidungsträger pro Kunde und potentiellem Kunde Schätzwerte über die an sie möglichen Verkäufe bei verschiedenen Verkaufsbesuchsstrategien unter Einbeziehung von Vergangenheitsdaten angibt. Diese kann er dann anhand einer vom Rechner errechneten Kurve über die daraus insgesamt zu erwartenden Verkäufe überprüfen; für Details , insbesondere bezüglich der Kurvenberechnung siehe (LODISH,1971),

In dem Beispiel treten Probleme sowohl in der Informationserstellung - Bestimmung von Verkaufsbesuchen - als auch ihrer Bewertung - Abschätzung der erzielbaren Verkäufe - auf. Lodish steht einem sinnvollen Einsatz von Modellen, mit deren Hilfe direkt aus den in einem Betrieb vorhandenen Daten, wie etwa den Verkäufen in vergangenen Planungsperioden, die benötigten Schätzwerte gewonnen werden können, ablehnend gegenüber[1]. Dementsprechend ist die von ihm vorgenommene Einbeziehung des Entscheidungsträgers (Vertreters) in den Informationsprozeß zwangsläufig: es liegt ein Problem vor, dessen Lösen - subjektiv gesehen - nicht programmierbar ist.

---

[1] Vgl. (LODISH,1971), p.26.

Der Zielbildungsprozeß hat im wesentlichen die Aufgabe, mögliche Handlungkonsequenzen im Sinne der verfolgten Zielgrößen zu bewerten, so daß anschließend eine Handlung ausgewählt werden kann, die einen größtmöglichen Zielerfüllungsgrad verspricht. Für eine rechnerunterstützte Lösung eines Entscheidungsproblems bedeutet dies insbesondere, daß für die Konsequenzen möglichst aller Handlungsalternativen eine Ordnungsrelation (Präferenzrelation[1]) existieren sollte, nach der der Rechner alle Handlungsalternativen entsprechend ihrer Konsequenzen anordnen[2] kann.

Existiert nun keine Präferenzrelation zur Anordnung der möglichen Handlungsalternativen, etwa aufgrund der subjektiven Einschätzung des Entscheidungsträgers, so kann der Rechner keine selbständige Anordnung mehr durchführen: der Entscheidungsträger selbst muß ein Teil des Zielsystems werden. Dies soll nun an einem einfachen betrieblichen Entscheidungsproblem, einem Personalzuordnungsproblem auf der Grundlage bekannter Zuordnungsmöglichkeiten, betrachtet werden.

Für eine Planungsperiode sollen n Arbeiter jeweils einem von m Aggregaten zugeordnet werden (n,m natürliche Zahlen). Entsprechend ihrer Qualifikation liegt fest, an welchen

---

[1] Die angesprochene Ordnungsrelation wird in der Entscheidungstheorie üblicherweise als Präferenzrelation bezeichnet; für eine Diskussion dieses Begriffes, insbesondere im Zusammenhang mit Bewertungsfunktionen, siehe (BAMBERG-COENENBERG, 1974), p.30ff., (GÄFGEN,1968), p.144ff., (HEINEN, 1966), p.147ff.

[2] Diese Anordnung muß nicht notwendig eine vollständige Ordnung aller Handlungsalternativen ergeben.

Aggregaten jeder der Arbeiter einsetzbar ist (sichere Information)[1]. Für eine Problemlösung eines solchen Zuordnungsproblems existieren Algorithmen[2], so daß die Entscheidung, welcher Arbeiter an welchem Aggregat unter dem Ziel einer Besetzung aller Aggregate einzusetzen ist,von einem Rechner auf der Grundlage der Einsetzbarkeitsdaten der einzelnen Arbeiter[3] getroffen werden kann. Tritt nun noch neben die Zielgröße "Besetzung aller Aggregate" die Zielgröße "persönliche und situationsabhängige Einschätzung einer Zuordnung durch den Entscheidungsträger", so ist, eben wegen der Persönlichkeits- und Situationsabhängigkeit, in der Regel eine programmierte Erstellung einer Präferenzrelation zur Bewertung einer Zuordnung unter beiden Zielgrößen nicht mehr möglich. Ein programmierter Zielbildungsprozeß ist damit nicht möglich.

Eine solche Situation liegt in (LOSBICHLER-MÜHLBACHER,1974) vor. Dort besteht die Möglichkeit, zu Beginn eines Zuordnungsprozesses einzelne Zuordnungen, etwa aufgrund von Ausbildungsüberlegungen, Motivation, u.ä., vorzugeben, wonach der Rechner einen Zuordnungsalgorithmus für das Restproblem ausführt. Eine daraus resultierende Zuordnung wird dann

---

[1] Für eine ausführliche Diskussion des Personalzuweisungsproblems innerhalb der Personaleinsatzplanung siehe (KLINGELHÖFER,1974).

[2] Für mögliche Algorithmen siehe (SIMMONARD,1966), p.341ff, (NOLTEMEIER,1976), p.125ff. sowie p.145ff.

[3] Zur Problematik des Erstellens solcher Daten, insbesondere wenn sie über die hier betrachtete Problemstellung hinaus die Eignung der einzelnen Arbeiter zum Bedienen der Aggregate einbeziehen sollen, siehe etwa (KLINGELHÖFER,1974).

jedoch nicht sofort als eine auszuführende Handlung betrachtet, sondern kann vom Entscheidungsträger dahingehend überprüft werden, ob "die erstellte Lösung in Bezug auf das Optimalitätsprinzip vertretbar ist"[1]. Diese Überprüfung soll neben einer Besetzung aller Aggregate persönliche, situationsabhängige Überlegungen des Entscheidungsträgers ("Erfahrung") miteinbeziehen. Dadurch wird - den obigen Betrachtungen entsprechend - die Programmierung einer Präferenzrelation, die diese Gesichtspunkte einbezieht, für eine Bewertung der möglichen Zuordnungen nahezu unmöglich. Folgerichtig wird daher in (LOSBICHLER-MÜHLBACHER,1974) der Entscheidungsträger in interaktiver Arbeitsweise ins Zielsystem der Zuordnungsproblemlösung eingeschlossen; er bringt dabei neben der vom Rechner überprüften Zielgröße, ob alle Aggregate besetzt werden können, seine persönliche Einschätzung einer Zuordnung als Bewertungskriterium in das Zielsystem ein.

Im vorhergehenden Beispiel war es nicht möglich, die subjektbezogene Zielgröße "Vertretbarkeit einer Zuordnung" in eine Präferenzrelation einzubeziehen. Eine weitere wichtige Möglichkeit, die ein Einbeziehen des Entscheidungsträgers in das Zielsystem erforderlich machen kann, ist ein Zielkonflikt[2]

---

[1] Vgl. (LOSBICHLER-MÜHLBACHER,1974), p.454.

[2] Zum Begriff Zielkonflikt siehe (HEINEN,1966), p.55ff., (BAMBERG-COENENBERG,1974), Teil 2.3.; für eine Diskussion der grundlegenden Lösungen eines Zielkonfliktes - Zieldominanz, Zielschisma und Zielkompromiß (hier im folgenden betrachtet) - siehe (HEINEN,1966), p.141ff.

zwischen verschiedenen, in einem Entscheidungsproblem verfolgten Zielen; "jedes Ziel determiniert in einem solchen Fall jeweils eine andere Alternative als optimal"[1]. Interaktive Arbeitsweise ist dann eine Möglichkeit[2], Handlungsalternativen anhand ihrer Konsequenzen trotz konfliktärer Ziele bezüglich eines subjektiven Zielerreichungsgrades einzustufen. Ein Beispiel dafür ist die Methode P.O.P., beschrieben in (BENAYOUN-TERGNY,1969); sie ermöglicht die Berechnung "zufriedenstellender" Lösungen für lineare Optimierungsprobleme mit mehreren, simultan zu optimierenden Zielfunktionen (vektorwertige Zielfunktion). Von vom Rechner berechneten Optimallösungen der einzelnen Optimierungsprobleme ausgehend hat der Entscheidungsträger dort die Möglichkeit, die Menge der Lösungsalternativen nach und nach zu verringern - nach jeder Veränderung bestimmt der Rechner die Menge der neuen Lösungsalternativen[3] -, bis er letztlich eine Lösung als zufriedenstellend akzeptiert (oder zumindest den Entscheidungsprozeß beendet).

Im Auswahlprozeß eines Entscheidungsprozesses können Probleme vor allem dann auftreten, wenn die Bewertung von

---

[1] Vgl. (HEINEN,1966), p.55.

[2] Für eine Übersicht über (interaktive) Methoden, mit denen Entscheidungen unter mehreren, ggf. konfliktären Zielen behandelt werden können - bei Information über alle Handlungsalternativen und deren Konsequenzen - siehe (ROY,1971), (FANDEL-WILHELM,1976).

[3] Für Details, insbesondere für mögliche Lösungsstrategien siehe (BENAYOUN-TERGNY,1969).

Handlungskonsequenzen entsprechend der Wertprämissen aus dem Zielsystem eine Menge von Handlungsalternativen als gleichwertig ausweist. In einem solchen Fall kann es sinnvoll oder notwendig sein, den Entscheidungsträger zum Auswählen einer Handlung heranzuziehen, die dann als Ergebnis des Entscheidungsprozesses angesehen werden soll. Da in betrieblichen Entscheidungsprozessen derartige Schwierigkeiten eine untergeordnete Rolle spielen (vgl. 1.2.2.1.), sollen sie hier nicht weiter betrachtet werden.

Zusammenfassend läßt sich sagen, daß in den einzelnen Phasen der Lösung eines betrieblichen Entscheidungsproblems (subjektiv) nicht programmierbare Problemlösungskomponenten insbesondere aufgrund der folgenden Probleme auftreten:

Informationsprozeß: Datenbeschaffungsprobleme und Datenbewertungsprobleme von Eingangs- und Zwischendaten;

Zielbildungsprozeß: Probleme bei der Formulierung von Zielgrößen, Probleme bei der Erstellung einer Präferenzrelation;

Auswahlprozeß : Probleme aufgrund unvollständiger Präferenzvorstellungen.

Sie entstehen auch dadurch, daß ein in der Realität gegebenes "ursprüngliches" Entscheidungsproblem durch zusätzliche Anforderungen an seine Lösungen erweitert wird. Solche Anforderungen berücksichtigen viele Ersteller (interaktiver)

Problemlösungen indirekt[1]. Ein Katalog von Anforderungen an eine rechnerunterstützte Problemlösung eines Entscheidungsproblems, der die Elemente

(1) einfache Modell- und Methodenvorstellungen für den Entscheidungsprozeß,

(2) weitgehender Ausschluß unsinniger Entscheidungsfindung,

(3) leichte Einflußnahme auf den Ablauf eines Entscheidungsprozesses,

(4) Adaptivität,

(5) Bereitstellung möglichst umfassender Entscheidungsmöglichkeiten trotz der in (1) geforderten "Einfachheit",

(6) leichte Kommunikationsmöglichkeit mit einem Rechner, insbesondere im Informationsprozeß,

wird von Little in (LITTLE,1970) unter dem Namen "decision calculus" beschrieben, der eine "model-based set of procedures for processing data and judgement to assist a manager in his decision making"[2] bezeichnen soll. Offenbar können die Forderungen (6) und (3) direkt interaktives Problemlösen bedingen. Aber auch weitere der oben angeführten Forderungen können interaktives Problemlösen bedingen; so kann etwa die Forderung (1) interaktiv zu lösende Probleme im Auswahlprozeß hervorrufen, wenn sie wegen der "Einfachheit" eine Präferenzrelation impliziert, die viele mögliche Handlungen

---

[1] Vgl. etwa die Problemlösungsentwicklung der in (MORTON, 1972) beschriebenen "Pionierinstallation" eines MES (Management-Entscheidungs-System).

[2] Vgl. (LITTLE,1970), p.470.

gleichwertig geeignet erscheinen läßt.

Neben die betrachteten Faktoren, die interaktives Problemlösen bei betrieblichen Entscheidungsproblemen bedingen können oder sinnvoll erscheinen lassen, treten in der Regel noch weitere Einflußgrößen, die in besonderem Maß durch die betriebliche Umwelt eines Entscheidungsproblems bedingt werden.

Darunter sind zu nennen[1]:

Verfügbare Hard- und Software,

Kostenvor- bzw. Nachteile bei Erstellung und Ausführung einer interaktiven Entscheidungsproblemlösung.

Jeder digitale Rechner verfügt über Möglichkeiten, mit ihm zu kommunizieren. Da jedoch - insbesondere bei großen Anlagen - dem Entscheidungsträger aus Auslastungs- und damit aus Kostengründen in der Regel nicht die gesamte reale Rechneranlage zur Verfügung gestellt werden kann, wird für interaktives Problemlösen im allgemeinen eine im Time-Sharing-Betrieb[2] arbeitende Rechenanlage[3] mit geeigneten Datenendgeräten[4] benötigt. Bezüglich Kosteneinflußgrößen und deren Bestimmung treten für interaktives Problemlösen die gleichen Probleme

---

[1] Vgl. (MERTENS-KRESS,1970), p.7ff., (SZYPERSKI-WINAND,1972), p.7ff.

[2] Zum Begriff Time-Sharing siehe etwa (RIBBECK,1973), (MEADOW,1970), chap.4.

[3] Für eine experimentelle Vergleichsstudie (auf der Basis von 1970) von Rechnern großer Hersteller (CDC,IBM,UNIVAC,u.a.) mit Time-Sharing-Betriebssystemen, betrachtet unter den Gesichtspunkten "Command Language and File System", "Editing Facilities" und Implementierung der Sprachen BASIC, FORTRAN und COBOL siehe (BORMANN-GRIMM,1972).

[4] Siehe dazu (MEADOW,1970), chap.2., (MARTIN,1973), sec. II und III.

auf, wie sie generell der Einsatz von Rechnern im Betrieb aufwirft[1]. Hinzu kommen dann noch interaktionsspezifische Kostenbetrachtungen[2], wie insbesondere Hardwarekosten und Kosten des von einem Rechner während eines Kommunikationsprozesses gebundenen Arbeitseinsatzes seines Benutzers, wodurch ebenfalls die Konzeption einer interaktiven Problemlösung für ein Entscheidungsproblem beeinflußt werden kann.

Ein besonderer Faktor schließlich, der auf die Konzeption einer Problemlösung Einfluß nehmen kann, ist die Zeit, in der Lösungsprozesse zur Lösung eines Problems ablaufen sollen. Man kann grob folgende Zeitvorstellungen für den zeitlichen Verlauf von Lösungsprozessen in einer Problemlösung unterscheiden:

a. ein Benutzer muß dann, wenn er Daten erhält, mit diesen unmittelbar einen Lösungsprozeß durchführen (etwa in einem Platzbuchungssystem), und/oder von einem Rechner erstellte Resultate sofort zur Verfügung haben (etwa, um sie innerhalb eines Entscheidungsprozesses zu beurteilen und beurteilt in den weiteren Entscheidungsprozeß einzubringen);

b. ein Benutzer ist nicht gezwungen, mit eingehenden Daten unmittelbar Lösungsprozesse auszuführen (etwa Veränderungen in der Kundenstammdatei eines Betriebes), und/oder er muß von einem Rechner gelieferte Resultate nicht notwendig

---

1) Für generelle Überlegungen zur Bewertung des Einsatzes von Rechnern in Betrieben siehe etwa (WILD,1971), (GROCHLA, 1970).

2) Für eine Betrachtung interaktionsspezifischer Kosten siehe (SZYPERSKI-WINAND,1972), (POENSGEN,1970), (MERTENS-KRESS, 1970).

sofort zur Verfügung haben (etwa müssen von einem Rechner erstellte Versanddokumente häufig nicht unmittelbar nach ihrer Erstellung weiterverarbeitet werden).

Die Zeitanforderung a. bedingt - unter der Voraussetzung, daß benötigte Betriebsmittel eines Rechners zur Verfügung stehen - zumeist eine Lösung des gegebenen Problems in einem "on-line"-Kommunikationsprozeß zwischen Benutzer und Rechner, also die Konzeption einer interaktiven Problemlösung. Hierbei ist zu sehen, daß die Anforderungen aus a. an die zeitliche Durchführung von Lösungsprozessen bei hinreichend weiter Auslegung der Begriffe unmittelbar und sofort die zeitlichen Anforderungen aus b. einschließen[1)]. Auf die damit verbundene Problematik einer Einbeziehung des Faktors Zeit in eine Modellbildung für interaktive Problemlösungen wird in 2.4. näher eingegangen.

Bisher wurde betrachtet, wie die einzelnen Komponenten eines betrieblichen Entscheidungsproblems (Informationsbeschaffung, Zielbildung und Auswahl) zusammen mit Einflußgrößen aus der betrieblichen Umwelt die Mitwirkung des Entscheidungsträgers in einem Lösungsprozeß bedingen oder sinnvoll erscheinen lassen. Es ist aber auch möglich, daß für ein Entscheidungsproblem Modelle und Methoden derart zusammengestellt werden

---

[1)] Dies deutet bereits an, daß es vom zeitlichen Charakter durchzuführender Lösungsprozesse her sinnvoll möglich ist, die Ausführung von Programmen ohne Benutzereinwirkung außer höchstens zu ihrem Starten als Grenz- oder Spezialfall interaktiven Problemlösens aufzufassen; dementsprechend wären dann Programme Grenz- oder Spezialfälle interaktiver Problemlösungen. Vgl. dazu hinten die Überlegungen zur Bedeutung von Korollar 2.3.1.1. sowie in 2.4.

können, daß ein Rechner unter Benutzung der betrieblichen Datenbasis (etwa einer Datenbank) ohne Benutzereingriff in den Entscheidungsprozeß Entscheidungen trifft[1], d.h. Handlungen auswählt. In diesem Fall ist die gesamte Problemlösung in einem Rechner implementiert. Der Fall schließlich, daß ein Entscheidungsträger allein ohne Mitwirkung eines Rechners im Entscheidungsprozeß Entscheidungen trifft, soll hier, wie in 1.2. erwähnt, nicht mitbetrachtet werden.

---

[1] Derart lösbare Entscheidungsprobleme werden auch als nicht "echte" Entscheidungsprobleme bezeichnet, vgl. (HEINEN, 1966), p.19.

1.3. Zur Verwendung interaktiver Entscheidungsproblemlösungen in Betrieben

Nachdem in 1.2.2.1. und 1.2.2.2. Bestimmungsgrößen betrieblicher Entscheidungsprobleme betrachtet worden sind, die die Konzeption einer rechnerunterstützten Problemlösung beeinflussen, soll nun abschließend die Verwendung interaktiver Entscheidungsproblemlösungen[1] in Betrieben erörtert werden.

Betriebliche Entscheidungsprobleme lassen sich schwerpunktmäßig aufgliedern[2] in

- strategische
- dispositive
- operative

Entscheidungsprobleme[3], die zu strategischen, dispositiven bzw. operativen Entscheidungen führen. Strategische Entscheidungen sind dabei Ziel- oder Zielsetzungsentscheidungen[4],

---

[1] Es sollen hier zunächst wieder wie in 1.2.2.2. interaktive Problemlösungen betrachtet werden, die für "on-line" ablaufende Kommunikationsprozesse konzipiert sind, vgl. Fußnote 1 auf p.21.

[2] Für ähnliche Dreiteilungen vgl. (GUTENBERG,1962), p.61,159, 160 (Führungsentscheidungen, dispositive und objektbezogene Entscheidungen), (ANSOFF,1966), Kap.1, (KIRSCH,1971a), p.19 (politische, administrative und operative Entscheidungen), (ZANGEMEISTER,1973), p.109; eine tiefere Betrachtung der einzelnen Begriffe sowie ihre Abgrenzung untereinander würde hier über den Rahmen der Arbeit hinausgehen.

[3] Diese Einteilung, die sich hier für die weiteren Betrachtungen als zweckmäßig erweist, ist eine mögliche Klassifizierung betrieblicher Entscheidungen; für eine Diskussion zur Typenbildung von Entscheidungen und eine auf dem Entscheidungsverhalten eines Entscheidungsträgers beruhende Typenbildung siehe (KAHLE,1973), p.93ff.

[4] Zum Begriff siehe (HEINEN,1966), p.18ff., (WÖHE,1973/74), p.109ff.

dispositive und operative Entscheidungen im wesentlichen Mittelentscheidungen[1]. Dispositive Entscheidungen sollen dabei in erster Linie zur Entwicklung von Programmen für die operative Ebene führen[2], während operative Entscheidungen im wesentlichen Entscheidungen sind, für die standardisierte Entscheidungsverfahren existieren. Gemessen an einer Typenbildung unternehmerischen Entscheidungsverhaltens - hier sei die Einteilung aus (KAHLE,1973) in reaktives, situatives, kreatives, konstitutives und teleologisch-novatives Entscheidungsverhalten zugrunde gelegt - kann man feststellen[3]: schwerpunktmäßig sind operative Entscheidungen durch reaktives und situatives, dispositive Entscheidungen durch kreatives und konstitutives und strategische Entscheidungen durch teleologisch-novatives Entscheidungsverhalten im Entscheidungsprozeß gekennzeichnet. In Anlehnung an Zangemeister[4] läßt sich dann die folgende Tabelle aufstellen[5]:

---

1) Zum Begriff Mittelentscheidung siehe (HEINEN,1966), p.19, (WÖHE,1973/74), p.109ff.

2) Vgl. (KIRSCH,1971a), p.20; ein Programm hat dabei allgemein die Bedeutung einer "Vorschrift"; vgl. auch (GUTENBERG, 1962), p.159.

3) Vgl. (KAHLE,1973), p.96ff.

4) Vgl. (ZANGEMEISTER,1973), p.109, Tabelle 1.

5) Zu ihr vgl. (SZYPERSKI,1971), p.50: "Das (Automatisierung, d.Verf.) mag für viele operationelle Routineprozesse zutreffen. Bei der Bewältigung dispositiver und vor allem strategischer Aufgaben kann demgegenüber auf die aktive Mitwirkung des Menschen nicht verzichtet werden."

	Operative Entscheidungs-probleme	Dispositive Entscheidungs-probleme	Strategische Entscheidungs-probleme
Erreichbare Formalisierung einer rechner-unterstützten Problemlösung	vollständig dem Rechner übertragene Problemlösung	adaptive und heuristische[1] dem Rechner übertragene sowie inter-aktive, nicht innovative[2] Problemlösung	interaktive innovative Problemlösung
Rolle des Ent-scheidungs-trägers dabei	reine Benutzer-funktion[3]	Gestaltungs-Kontroll- und reine Benutzer-funktion, mehrfach funktionales Element	mehrfach funktionales, i.d.R. die Lösungsstrate-gie führendes Element

Figur 1.3.1.

In jeder der drei Problemklassen treten strukturierte und nicht- oder nur teilstrukturierte Probleme auf[4], wobei operative Entscheidungsprobleme schwerpunktmäßig als strukturiert und strategische als unstrukturiert angesehen werden können[5].

---

[1] Zu den Begriffen heuristisch und adaptiv vgl. Fußnote 2 auf p.9.

[2] Zum Begriff innovatives Entscheiden siehe (KIRSCH,1971), p.144ff., (KIESER,1973), p.57/58.

[3] Im Sinne von "benutzen von vom Rechner gelieferten Resultaten , ohne Einfluß auf ihre Erstellung zu haben".

[4] Zum Begriff strukturiertes bzw. nicht strukturiertes Problem siehe (KIRSCH,1971), p.142ff.

[5] Vgl. (ZAHN,1973), p.599ff.; dort finden sich auch einige Beispiele (Tabelle 1.) für mehr oder weniger stark strukturierte Entscheidungsprobleme.

Operative Entscheidungen können häufig als Routineentscheidungen durch standardisierte Arbeitsverfahren von einem Rechner getroffen werden, wenn die notwendigen Daten im Betrieb vorhanden sind - etwa im Rahmen einer umfangreichen integrierten Datenerfassung[1]. Beispiele dafür sind vor allem solche, der ausführenden Ebene eines Betriebes nahe Entscheidungsprobleme, die relativ einfach durch Modelle beschrieben werden können, und für die geeignete, ohne zu großen Aufwand rechenbare Methoden zur Berechnung von Lösungen existieren - etwa aus der linearen Optimierung, einfache Simulationstechniken, u.ä.. In (MERTENS,1972) und (MERTENS-GRIESE,1972) sind - innerhalb der einzelnen Betriebsfunktionen - einige Anwendungen aus der betrieblichen Praxis beschrieben.

Während operative und strategische Entscheidungsprobleme relativ deutlich voneinander unterschieden werden können[2], haben dispositive Entscheidungsprobleme nach beiden Seiten hin "fließende Grenzen". Dies spiegelt sich auch darin wider, daß sich für ihr rechnerunterstütztes Lösen kaum ein Schwerpunkt angeben läßt, sondern die gesamte Palette rechnerunterstützter Problemlösungsmethoden verwandt wird. Interaktive Problemlösungen von dispositiven Entscheidungsproblemen zurechenbaren Problemen werden etwa in (MEVERT-DICKSON,1973)

---

[1] Zum Begriff intergrierte Datenverarbeitung, der integrierte Datenerfassung einschließt, siehe (MERTENS,1972), 1.1.-1.5.

[2] Vgl. (ZAHN,1973), p.602.

(Produktionsprogrammplanung), (MORTON,1972) (Produktions- und Absatzplanung), (LITTLE,1973) (Absatzplanung) und (EMMERT,1975) (Investitionsplanung) beschrieben[1].

Strategische Entscheidungsprobleme sind in der Regel komplexe schlecht strukturierte Probleme, die innovative Entscheidungen erfordern. Sollen sie rechnerunterstützt derart gelöst werden, daß ein Rechner nicht nur eine Entscheidungsfindung durch Liefern "möglichst guter" Ausgangsinformationen[2] erleichtern, sondern im Rahmen von Betriebs(teil)modellen auch Handlungsalternativen, deren Konsequenzen u.ä. zur Diskussion stellen soll, so ist interaktives Problemlösen nahezu zwangsläufig. Interaktive Problemlösungen für das Treffen strategischer Entscheidungen werden etwa beschrieben in (CARTER-COHEN,1973)[3] (Vergleich von langfristig geplanten Projekten), (HAMILTON-MOSES,1974) (Finanzplanung).

Einen Überblick über die derzeit auf dem deutschen Software-Markt angebotenen rechnerunterstützten Problemlösungen betrieblicher Probleme bieten der "ISIS-Software-Report"[4] sowie

---

[1] Für weitere Beschreibungen betrieblicher Anwendungen siehe etwa (MERTENS-GRIESE,1972).

[2] Das "Liefern möglichst guter Ausgangsinformationen" ist die Hauptaufgabe von Management-Informations-Systemen (MIS), auf die hier jedoch nicht weiter eingegangen werden soll; für eine Definition eines MIS, sowie seinen Aufbau und Verwendung siehe (KOREIMAN,1971) (Definition: p.21), (DWORATSCHEK,1971) u.a..

[3] Siehe dort p. 503ff. für eine Diskussion weiterer rechnerunterstützter Problemlösungen betrieblicher Entscheidungsprobleme.

[4] Vgl. (ISIS,1976).

die Publikationen der Hersteller von Rechenanlagen[1]; daneben steht eine große Anzahl von "Eigenproduktionen" von Betrieben, die nicht auf dem Software-Markt angeboten werden.

Generell läßt sich feststellen, daß in der Bundesrepublik Deutschland ein Trend von (vorhandenen) Problemlösungen operativer Entscheidungsprobleme zu rechnerunterstützten Problemlösungen dispositiver und strategischer Entscheidungsprobleme zu beobachten ist, während etwa in den USA eine umgekehrte Tendenz vorhanden ist[2].

Neben dem interaktiven Lösen betrieblicher Entscheidungsprobleme werden auch innerhalb der folgenden wesentlichen, in Betrieben wahrgenommenen Aufgaben interaktive Arbeitsweisen verwandt:

- rechnerunterstützte Schulung[3],
- interaktive Programmentwicklung[4],
- interaktives Konstruieren[5].

---

[1] Beispiele dafür sind etwa die "IBM-Nachrichten" der Firma IBM, die von der Firma SIEMENS herausgegebene "OR-Praxis" die Zeitschrift "Datascope" der Firma UNIVAC und andere Publikationen sowie entsprechende Kataloge der Hersteller.

[2] Vgl. (GROCHLA,1971), p.225.

[3] Zum derzeitigen "state-of-the-art" siehe etwa (BRUNNSTEIN-HÄFNER-HÄNDLER,1974); vgl. auch (MEADOW,1970), chap.8, (MARTIN,1973), chap.24., (GLATTHAAR,1974), Teil II und IV.

[4] Siehe dazu etwa (KUPKA-WILSING,1975), Teil 1., (MEADOW, 1970), chap.10 ; speziell etwa (HOFFMANN-WINCKLER,1973).

[5] Zu grundlegenden Fragen interaktiven Konstruierens siehe (JOHNSON,1968); siehe auch (MEADOW,1970), chap.11.

Bei allen Anwendungen interaktiver Arbeitsweisen in Betrieben ist neben den von betrieblichen Gegebenheiten ausgehenden Einflüssen auf die Konzeption von Problemlösungen auch stets das Verhalten eines Benutzers in eine Problemlösungskonzeption einzubeziehen[1]. Dabei sind allgemeine Verhaltensaspekte eines Benutzers in interaktiven Problemlösungsprozessen zu berücksichtigen, die unter betriebsspezifischen Gesichtspunkten noch besondere Ausprägungen erfahren können[2]. Einige wesentliche Aspekte sind[3]:

1. Berücksichtigung der allgemeinen geistigen Fähigkeiten eines Menschen, insbesondere des kurzfristigen Speicher-

---

[1] Vgl. (MARTIN,1973), p.309: "It is very important, that careful attention be devoted to user psychology as more and more people will be confronted with terminals in their daily living"; vgl. die Betonung der Betrachtung von offenen Modellen des Entscheidungsverhaltens in (KIRSCH, 1970).

[2] Vgl. etwa die Untersuchung von (BRADY,1967), daß das "senior management" in den USA - zum Zeitpunkt der Studie - kaum direkten Kontakt zu Rechnersystemen hatte, sowie (MARTIN,1973), p.437ff., der einen Weg zur Überwindung der Lücke zwischen Management und Rechner - "In many organisations it would be unthinkable for high management, to use computer terminals directly" (p.438) - in der Einrichtung von "data secretaries" zwischen Management und Rechner sieht. Vgl. auch die Überlegungen in (KIESER-KUBICEK,1974) zur Organisationsstruktur eines MIS, in denen neben einer betriebsspezifischen Einbettung auch die "Akzeptanz" eines MIS durch seine zukünftigen Benutzer gefordert wird; vgl. auch (KIESER,1973), p.140ff.

[3] Eine ausführliche Behandlung des Benutzerverhaltens würde über den Rahmen dieser Arbeit hinausgehen; vgl. jedoch die Betrachtungen zu Modellen des Benutzerverhaltens in 2.3.1.

vermögens von Informationen[1], wodurch ein allgemeiner Rahmen für eine Bestimmung von Schwächen und Stärken eines Benutzers (vgl. 1.2.1.) gegeben wird;

2. Aufbau, Erhaltung bzw. Förderung von Lösungsmotivation eines Benutzers[2] und eine damit ggf. verbundene Leistungsverbesserung durch
   - einen "sinnvoll" gestalteten[3] Kommunikationsprozeß, der Unter- und insbesondere Überforderung (sowohl in psychischer[4] wie physischer[5] Sicht) vermeidet,
   - eine Lösungsprozesse nicht störende sondern fördernde Antwortzeit[6],

---

1) Dazu siehe (KIRSCH,1970), p.76ff., (NEWELL-SIMON,1972), p.791ff., (MARTIN,1973), p.335ff., (KIESER,1973), p.44ff.

2) Vgl. die Meinung von (KIESER,1973), p.299 - bezogen auf MIS - "Offenbar finden die Ansätze der Entscheidungs- und Informationsverarbeitungstheorie bei den Betriebswirten und Informatikern weit mehr Anklang, als die gegenwärtig nur relativ vagen, noch wenig faßbaren motivationstheoretischen Ansätze, die sich sehr schlecht in die durch Rationalprinzip und formale Entscheidungslogik geprägten Denkkategorien einfügen lassen".

3) Siehe dazu (MARTIN,1973), p.311ff., (MERTENS-GRIESE,1972) p.102ff.; vgl. auch die Forderung nach "leichter Kommunikation" mit einem Rechner im "decision calculus" von (LITTLE,1970).

4) Vgl. auch den ersten Aspekt.

5) Hierbei treten etwa Probleme beim interaktiven Arbeiten von Datensichtstationen aus auf, wenn ergonometrische Forderungen - vgl. dazu (HULTGREN,1975) - nur unzureichend berücksichtigt werden.

6) Vgl. (MARTIN,1973), p.321ff., (NICKERSON,1969), p.170ff.; es ist jedoch zu sehen, daß nicht nur bei interaktivem Problemlösen, das "on-line" erfolgt, Antwortzeitprobleme vorhanden sind, sondern daß auch ein mit einem etwa im Stapelbetrieb laufenden Rechner arbeitender Benutzer eine lange "Antwortzeit" als hemmend empfinden kann; vgl. auch Teil 2.4. der Arbeit.

- Abstimmung einer interaktiven Problemlösung auf ihren Benutzerkreis[1].

Untersuchungen zu Problemen des Benutzerverhaltens in interaktiven Lösungsprozessen sind etwa in (NICKERSON,1969)[2], (SACKMAN,1970)[3], (EDSTRÖM,1973) und (EASON-DAMODARAN-STEWART, 1975) enthalten; von einer Theorie des Benutzerverhaltens in interaktiven Problemlösungsprozessen kann jedoch derzeit noch nicht gesprochen werden[4].

Nachdem im ersten Teil der Arbeit interaktives Problemlösen anhand betrieblicher Entscheidungsprobleme analysiert worden ist, wird nun im zweiten Teil - von bestehenden interaktiven Problemlösungen ausgehend - ein Modell abgeleitet, das wesentliche Merkmale interaktiver Problemlösungen einschließt.

---

[1] Darunter fallen etwa das Überspringen von Lösungsschritten für einen geübten Benutzer sowie die Verwendung von "Hilfefunktionen" für einen ungeübten Benutzer, vgl. (NICKERSON, 1969), p.168.

[2] In (NICKERSON,1969) finden sich zahlreiche Literaturhinweise auf frühere derartige Untersuchungen.

[3] In (SACKMAN,1970) wird insbesondere "on-line" gegen "off-line" Problemlösen betrachtet.

[4] Vgl. (SACKMAN, 1970), p.3: "Although the enhancement of the human intellekt with computers has been the subject of much speculation, remarkably little solid scientific work has been done in man-computer problem-solving"; vgl. auch (KUPKA-WILSING,1975), p.41: "Die psychologischen Bedingungen für den Dialog des Menschen mit einem Rechner sind noch weitgehend unerforscht".

2. Beschreibungen interaktiver Problemlösungen durch Modelle

Anhand ausgewählter betrieblicher, rechnerunterstützt zu lösender Entscheidungsprobleme ist betrachtet worden, wie eine Problemstellung aufgrund der Möglichkeiten und Grenzen eines Rechners und Benutzers eine interaktive Problemlösung bedingen oder sinnvoll erscheinen lassen kann. Damit sind bereits gewisse Aussagen über das Aussehen einer interaktiven Problemlösung verbunden. Bevor nun darauf aufbauend interaktive Problemlösungen allgemein charakterisiert werden sollen, wird eine interaktive Problemlösung eines Entscheidungsproblems aus der Versanddisposition eines Betriebes entwickelt, die im wesentlichen die folgenden Aufgaben haben soll:

1. Liefern von Beschreibungsmerkmalen (2.1.2.), die dann in 2.2. unter allgemeineren Gesichtspunkten betrachtet und erweitert werden;
2. Veranschaulichung des in 2.3.1. definierten Modells (2.3.2.) und dessen Eigenschaften (3.1.2., 3.2.2. und 3.4.3.).

2.1. Interaktives[1] Lösen einer Transportmitteldisposition

Es wird von der folgenden Situation eines Betriebes B ausgegangen: B stellt ein Gut G - gemessen in Mengeneinheiten ME - in Vorratsfertigung her; im Rahmen der in B bislang rechner-

---

[1] Interaktiv schließe hier zunächst - wie in 1.2.2.2. - noch "on-line" ein.

unterstützt gelösten Probleme existiert eine Datenbank[1], die u.a. folgende Daten für jeweils eine Planungsperiode τ enthält (auf einen Index τ wird im folgenden verzichtet) :

Lieferadressen von n Kunden $K_1,\ldots,K_n$, Bestellmengen $k_1,\ldots,k_n$ der Kunden (ggf. gleich Null), Entfernungen $e_1,\ldots,e_n$ (in EE) der Kunden von B;	innerhalb einer Kundenstammdatei
konstante Kostensätze (in GE/ME·EE) $p_1,\ldots,p_m$ von m (betriebseigenen und -fremden) Transportmitteln $T_1,\ldots,T_m$ und deren Kapazitätsschranken $b_1,..,b_m$;	innerhalb einer Transportmittelstammdatei

GE bezeichne Geldeinheiten, EE Entfernungseinheiten, m,n natürliche Zahlen, $m \geq 1, n \geq 1$; die Disposition der Auslieferung von G[2] an Kunden, die in τ zu beliefern sind, wird bislang nicht rechnerunterstützt durchgeführt; die Vertriebskosten, darunter insbesondere die Transportkosten von G haben einen wesentlichen Anteil an seinen Gesamtkosten; eine Kostenanalyse hat ergeben[3], daß eine Minimierung der Trans-

---

[1] Zum Begriff siehe (WEDEKIND,1975), insbesondere Teil 1.1. und 1.2.

[2] Dieses Problem kann als ein dispositives Entscheidungsproblem angesehen werden, vgl. (GUTENBERG,1962), p.159, wo "Arbeitsverteilung" als dispositives Entscheidungsproblem betrachtet wird.

[3] Zur Problematik solcher Betrachtungen siehe (WILD,1971).

portkosten durch rechnerunterstützte Auswahl von Transportmitteln (aus $T_1,\ldots,T_m$ pro Planungsperiode) erreicht werden kann, wobei nicht unwesentlich ist, daß dafür benötigte Daten in der Datenbank bereits zur Verfügung stehen[1]. Neben einer Senkung der Gesamtkosten (pro ME) von G[2] ist eine rechnerunterstützte gesamt-transportkostenminimale Transportmittelauswahl ein Schritt in Richtung auf eine umfassende integrierte Datenverarbeitung[3] von B.

2.1.1. Die Problemstellung und eine Problemlösung

Durch die oben geschilderte Situation fällt für B in jeder Planungsperiode τ das Entscheidungsproblem P an, mit den (eigenen und angebotenen fremden) Transportmitteln die Bestellmengen $k_1,\ldots,k_n$ in dem Sinne kostenoptimal auszuliefern, daß die Gesamttransportkosten $\sum_{i,j} p_i e_j x_{ij}$[4] minimal werden, wobei x_{ij} die in τ mit Transportmittel T_i zum Kunden K_j transportierte Menge von G in ME bezeichnet. Es wird vorausgesetzt, daß B stets in der Lage ist, von einem Lager die bestellten Mengen von G zu liefern, und daß G so beschaffen ist, daß pro Fahrt eines Transportmittels aus $T_1,\ldots,T_m$ stets nur ein

---

1) Vgl. (WILD,1971), p.319: "Die Anwendung der allgemeinen Wirtschaftlichkeitsmaxime auf Informationsverarbeitungsprozesse führt ... zu dem Postulat, eine Information so selten wie nötig (möglichst nur einmal) zu produzieren und sie so oft wie möglich zu verwenden."

2) Dabei entstehende Kosten durch Programmerstellung oder -miete, Rechenzeitverbrauch - auch bei interaktiver Arbeitsweise - , u.ä. seien berücksichtigt.

3) Zur integrierten Datenverarbeitung siehe (MERTENS,1972), p.23ff.

4) Die (leere) Rückfahrt nach B sei in den Kostensätzen eingeschlossen.

Kunde beliefert werden kann.

Zusätzlich zu dem Ziel "Kostenminimalität" wird von der Versanddisposition, in deren Aufgabenbereich das Lösen von P fällt[1], gefordert, daß es möglich sein soll:

a. zu transportierende Mengen x_{ij} "leicht", vgl. 1.2.2.2.(6), vorzugeben, um dadurch von vorn herein Kunden durch bestimmte der angebotenen Transportmittel beliefern lassen zu können;

b. eine "möglichst gute" Transportmittelverteilung zu finden, falls innerhalb von τ kein Transport aller bestellten Mengen mit den Kapazitäten der einsetzbaren Transportmittel möglich ist, vgl. 1.2.2.2.(5).

Ohne Berücksichtigung der beiden Forderungen a. und b. läßt sich P modellhaft als das folgende lineare Optimierungsproblem (ein Transportproblem) (P) formulieren:

$$\text{minimiere } f(x_{11},\ldots,x_{mn}) := \sum_{i,j=1}^{m,n} p_i e_j x_{ij}$$

$$\text{unter den Nebenbedingungen } \sum_{i=1}^{m} x_{ij} = k_j,\ j=1,\ldots,n\ ;$$

$$\sum_{j=1}^{n} x_{ij} \leq b_i,\ i=1,\ldots,m\ ;$$

$$\text{und sinnvollerweise } x_{ij} \geq 0 \text{ für alle } i \text{ und } j.$$

Es ist offenbar ebenfalls sinnvoll, $b_i > 0$ und $p_i > 0$ für $i=1,..,m$ und $e_j \geq 0$ und $k_j \geq 0$ für $j=1,..,n$ vorauszusetzen;

---

[1] Vgl. (MERTENS,1972), p.75ff.

[2] Zur Bedeutung möglichst guter Lösungen für betriebliche Distributionsaufgaben vgl. (BLOECH-IHDE,1972), p.108: "Die physische Produktdistribution ...ist integrativer Bestandteil der unternehmerischen Leistungsverwertung."

weiterhin seien b_i, p_i, e_j und k_j nach oben beschränkt für alle i und j.

Eine Problemlösung von (P) ist dann mit einem geeigneten Transportalgorithmus[1] - er wird im folgenden TPL (für Transport-Problem-Algorithmus) genannt - angebbar.

Weiterhin müssen nun noch die Ziele a. und b. in einer Problemlösung des Transportmitteldisposition-Entscheidungsproblems berücksichtigt werden. Die Erfüllung von Forderung a. beinhaltet dabei insbesondere, daß der Entscheidungsträger auf möglichst einfache Weise über die zu vergebenden Transporte informiert wird und aus Vergaben von Transporten x_{ij} resultierende Datenveränderungen leicht überblicken kann; beides läßt interaktive Arbeitsweise zur Erfüllung von Forderung a. sinnvoll erscheinen.
Berücksichtigung der Zielsetzung b. in einer Problemlösung bedeutet die Berücksichtigung so subtiler Größen, wie das Ausmaß an Verärgerung eines Kunden, "Good-will-Verlust", u.ä., aufgrund von nicht oder nur teilweise erfolgter Belieferung. Setzt man - subjektiv vom Entscheidungsträger her gesehen - voraus, daß eine Berücksichtigung solcher Größen nicht in einem Modell erfaßbar ist, so bedingt die Zielsetzung b., den Entscheidungsträger an den Stellen einzube-

---

[1] Für ein Buch aus der reichen Auswahl an Literatur, die sich mit dem Lösen linearer Optimierungsprobleme befaßt, siehe etwa (SIMMONARD,1966); zu Transportalgorithmen vgl. dort p.244ff.; für Algorithmen auf graphentheoretischer Basis vgl. (NOLTEMEIER,1976), p.138ff.

ziehen, an denen Forderung b. einen Transportmitteldispositionsprozeß beeinflußt. Daneben kann Forderung b. auch teilweise programmiert berücksichtigt werden, etwa durch Einbeziehung von Kunden ermöglichter Teillieferungen, sofern die dafür benötigten Daten - Strafkosten, Verzögerungskosten u.ä. - in der Kundenstammdatei verfügbar sind.

Bekanntlich[1] existiert eine zulässige Lösung von (P) genau dann, wenn

$$\sum_{i=1}^{m} b_i \geq \sum_{j=1}^{n} k_j$$

gilt, sowie eine optimale ganzzahlige Lösung, falls b_i und k_j zusätzlich ganzzahlig sind[2]; i=1,...,m, j=1,...,n.

Dann beschreibt der in Figur 2.1.1.1. mit den Sinnbildern für Datenfluß und Programmablauf nach DIN 66001[3] gegebene Ablaufplan eine mögliche Problemlösung für das Transportmitteldispositions-Entscheidungsproblem.

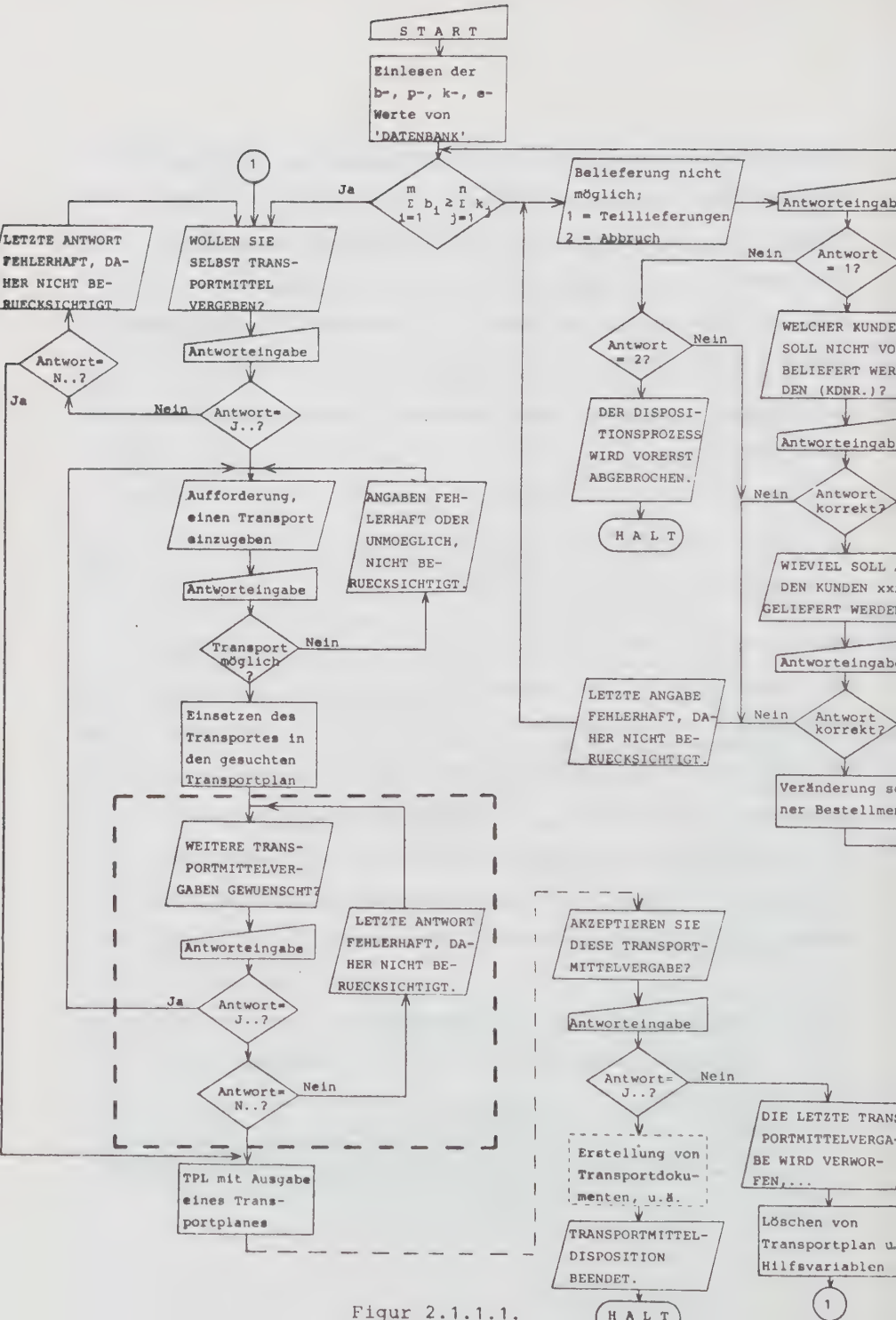
Entsprechend der Tatsache, daß außer Programmteilen auch Aktivitäten eines Benutzers zu betrachten sind, werden im Ablaufplan von Figur 2.1.1.1. Programmablaufplan- und allgemeine Datenflußsymbole benutzt. Die Dokumentation einer

---

[1] Vgl. (SIMMONARD,1966), p.226, Theorem 11.1

[2] Vgl. (SIMMONARD,1966), p.229, Korollar zu Theorem 11.3.

[3] Vgl. (DIN,1969).


START
Einlesen der b-, p-, k-, e-Werte von 'DATENBANK'
1
Ja
m n
Σ b_i ≥ Σ k_j
i=1 j=1
Belieferung nicht möglich;
1 = Teillieferungen
2 = Abbruch
Antworteingab
Antwort = 1?
Nein
Antwort = 2?
Nein
DER DISPOSITIONSPROZESS WIRD VORERST ABGEBROCHEN.
HALT
WELCHER KUNDE SOLL NICHT VO BELIEFERT WER DEN (KDNR.)?
Antworteingab
Antwort korrekt?
Nein
WIEVIEL SOLL DEN KUNDEN xx GELIEFERT WERDE
Antworteingab
Antwort korrekt?
Nein
LETZTE ANGABE FEHLERHAFT, DAHER NICHT BERUECKSICHTIGT.
Veränderung s ner Bestellme
LETZTE ANTWORT FEHLERHAFT, DAHER NICHT BERUECKSICHTIGT.
WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN?
Antworteingabe
Antwort= J..?
Nein
Antwort= N..?
Ja
Aufforderung, einen Transport einzugeben
Antworteingabe
Transport möglich ?
Nein
ANGABEN FEHLERHAFT ODER UNMOEGLICH, NICHT BERUECKSICHTIGT.
Einsetzen des Transportes in den gesuchten Transportplan
WEITERE TRANSPORTMITTELVERGABEN GEWUENSCHT?
Antworteingabe
Antwort= J..?
Ja
Antwort= N..?
Nein
LETZTE ANTWORT FEHLERHAFT, DAHER NICHT BERUECKSICHTIGT.
TPL mit Ausgabe eines Transportplanes
AKZEPTIEREN SIE DIESE TRANSPORTMITTELVERGABE?
Antworteingabe
Antwort= J..?
Nein
Erstellung von Transportdokumenten, u.ä.
TRANSPORTMITTELDISPOSITION BEENDET.
HALT
DIE LETZTE TRANSPORTMITTELVERGABE WIRD VERWORFEN,...
Löschen von Transportplan u. Hilfsvariablen
1

Figur 2.1.1.1.

Implementierung auf dem Rechner UNIVAC 1108[1] der in Figur 2.1.1.1. skizzierten Lösungskomponenten wird in Anhang 1 vorgenommen[2].

In Figur 2.1.1.1. werden mögliche Kommunikationsprozesse zwischen Benutzer und Rechner bereits im Ablaufplan durch das - nicht notwendig direkte - Aufeinanderfolgen der Symbole

deutlich. Das obere Symbol umschließt Ausgabeaktivitäten des Rechners, das untere Eingabeaktivitäten eines Benutzers. Eine abwechselnde Folge beider kennzeichnet dabei mögliche Elemente eines Kommunikationsprozesses, mit dessen Hilfe - neben Berechnungen des Rechners und Überlegungen des Benutzers - ein Lösungsprozeß für eine Transportmitteldisposition durchgeführt werden kann. Eine Sonderstellung hat

---

[1] Die Implementierung erfolgte auf dem Rechner UNIVAC 1108 der Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen unter dem Time-Sharing Betriebssystem EXEC 8.

[2] Ähnliche Programmkomponenten, wie die in Anhang 1. beschriebenen sind in Programmen enthalten, die zur Automatisierung und Optimierung betrieblicher Transportaufgaben auf dem Software-Markt angeboten werden, vgl. etwa COVER (Coopers Vehicle Routing) von Honeywell-Bull.

der Start von Lösungsprozessen in Figur 2.1.1.1., da er ebenfalls eine Benutzeraktivität darstellt. Um Problemlösungsstop - vom Rechner her gesehen - in einem Lösungsprozeß in Figur 2.1.1.1. zu erreichen, ist eine Folge von (nicht notwendig "sinnvollen") Benutzer- und Rechneraktivitäten erforderlich; für eine exakte Behandlung dieser Überlegung siehe 3.4.

2.1.2. Beschreibungsmerkmale dieser Problemlösung

An der in Figur 2.1.1.1. skizzierten Problemlösung und in ihr führbaren Lösungsprozessen sind einige, für das folgende interessante Beschreibungsmerkmale festzustellen[1]:

(i) die den jeweils "nach außen sichtbaren" Kommunikationsprozeß darstellenden Aktivitäten von Benutzer und Rechner können - abstrakt gesehen - als (zeitlich) aufeinanderfolgende Elemente einer Folge von Zeichenketten angesehen werden;

(ii) die (statische) Beschreibung der Problemlösung besteht aus einer Beschreibung der möglichen Kommunikation sowie den "intern" vom Rechner durchgeführten Operationen;

---

[1] Zum "äußeren Aussehen" solcher Lösungsprozesse vgl. Anhang 2.

(iii) ein Benutzer hat für seine Aktivitäten gewisse Wahlmöglichkeiten zur Verfügung, etwa Wertmodifikationen, Ja-Nein-Entscheidungen, u.ä.;

(iv) ein Benutzer beeinflußt einen Lösungsprozeß mit seinen Aktivitäten (Eingabe von Daten) auf zwei sich nicht ausschließende Weisen:

(iv)1. er wirkt auf die Folge der in einem Lösungsprozeß durchzuführenden Lösungsschritte (Wege im Graphen des Ablaufplanes aus Figur 2.1.1.1.) ein; vgl. etwa die Fortsetzungsmöglichkeiten von Lösungsprozessen aufgrund von Ja-Nein-Entscheidungen in Figur 2.1.1.1.;

(iv)2. er beeinflußt Werte von Variablen, die in die Berechnung von Werten für Ausgabevariable[1] "einfließen"; vgl. etwa das Geschehen nach korrekten Antworteingaben in Figur 2.1.1.1.;

(v) die möglichen Lösungsprozesse sind - auf den Rechner bezogen - sequentiell in dem Sinne, daß Benutzer und Rechner abwechselnd Aktivitäten in einen Kommunikationsprozeß einbringen[2];

(vi) Benutzereingaben in alphanumerischer Form an einer der vorprogrammierten Stellen für Eingriffsmöglichkeiten

---

[1] Zum Begriff Ausgabevariable in programmtheoretischer Sicht siehe 3.3.

[2] Dies schließt (zunächst) nicht aus, daß "intern" vom Rechner auszuführende Operationen - geeignet synchronisiert - parallel durchgeführt werden können, oder ein Benutzer parallel zur Arbeit des Rechners überlegt; darauf wird in 2.2.1. näher eingegangen; vgl. auch (KUPKA-WILSING, 1975), p.43.

bewirken, daß ein Lösungsprozeß an dieser Stelle in eindeutiger Weise fortgesetzt werden kann; vgl. etwa in Figur 2.1.1.1. die Wahl aus dem "Menü", das aus den Zahlen 1 und 2 sowie - indirekt - beliebigen weiteren alphanumerischen Eingaben besteht, falls die Kapazitäten der benötigten Transportmittel nicht zum Transport aller bestellten Mengen ausreichen.

Die hier aufgeführten Punkte werden nun im Zusammenhang mit weiteren, in der Literatur beschriebenen interaktiven Problemlösungen benutzt, um allgemeine Beschreibungsmerkmale interaktiver Problemlösungen zu gewinnen, die Grundlage der anschließend durchgeführten Modellbildung werden.

2.2. Allgemeine Beschreibungsmerkmale interaktiver Problemlösungen

Betrachtet man neben der in 2.1.1. beschriebenen interaktiven Transportmitteldisposition in der Literatur dokumentierte interaktive Problemlösungen[1], so lassen sich daraus folgende Beschreibungsmerkmale ableiten, die zum Teil schon in 2.1.2. festgestellt wurden:

(i) eine Folge von Benutzer- und Rechneraktivitäten in einem Kommunikationsprozeß läßt sich formal als eine Folge von Worten über einem geeigneten Alphabet[2] darstellen[3]; vgl. 2.1.2.(i).

---

[1] Für eine Auswahl betrachte man etwa die in 1.2.2.2. und 1.3. angeführten interaktiven Problemlösungen, sowie Beispiele in (MARTIN,1973), (MEADOW,1970), (RIBBECK,1973) u.a.. Die dabei getroffene Auswahl ist als rein willkürlich anzusehen. Dies ist jedoch unter der in 2.2. verfolgten Zielsetzung, aus existierenden interaktiven Problemlösungen deduktiv allgemeine Beschreibungskriterien abzuleiten, durchaus sinnvoll, da die unüberschaubare Menge von in der Literatur beschriebenen interaktiven Problemlösungen eine Stichprobenauswahl für eine "repräsentative" Stichprobe unter statistischen Gesichtspunkten ausschließt.

[2] Für die Begriffe "Alphabet" und "Wort" siehe (HOPCROFT-ULLMAN, 1969); derartige Wortfolgen können sich auch aus Bildsymbolen, Schalterstellungen u.ä. zusammensetzen; letztere sind von besonderem Interesse bei "in die Hardware eingebauten Dialogelementen" ((MARTIN,1973),p.143ff.), die bei Datenendgeräten für besondere Anwendungen, etwa "Buchungsterminals", "APL-Terminals" auftreten. Auch bei gesprochener Kommunikation - soweit überhaupt realisierbar - zwischen Benutzer und Rechner kann die Annahme aufrecht erhalten werden.

[3] Vgl.(KUPKA,1972), p.158, (KUPKA-WILSING,1975), p.42/43: "Wenn wir von der physikalischen Natur der Äußerungen (von Rechner und Benutzer d. Verf.) absehen, so können wir diese als Zeichen oder Zeichenketten auffassen". Für Beispiele siehe etwa neben Anhang 2 in (LODISH,1971), "Table 1", Beispieldarstellungen in (RIBBECK,1973), (MARTIN,1973) - chap. 15. für Benutzung von Bildsymbolen -, u.a.

(ii) Ein Lösungsprozeß setzt sich aus einem Kommunikationsprozeß, "intern" vom Rechner durchgeführten Berechnungen sowie Denkprozessen eines Benutzers zusammen, wobei über letzte kaum eine explizite Aussage gemacht wird (und derzeit wohl auch kaum gemacht werden kann[1]);

(iii) einem Benutzer stehen zwei verschiedene Möglichkeiten, Aktivitäten in einen Lösungsprozeß einzubringen, zur Verfügung:

(iii)1. der Problemlösungsablauf sieht Stellen vor, an denen der Rechner Aktivitäten des Benutzers erwartet; darunter fällt auch, wenn ein Problemlösungsprozeß durch einen Benutzer gestartet wird, was wohl der Regelfall, aber nicht notwendig ist. Beispiele für solche Stellen, die in (KUPKA-WILSING,1974) "schwache Unterbrechungen" genannt werden, sind etwa Anfragen des Rechners nach benötigten Daten und Bereitstellung eines "Menüs" von möglichen Problemlösungskomponenten durch den Rechner, aus dem ein Benutzer eine Auswahl zur Bestimmung des Fortganges eines Problemlösungsprozesses treffen kann;

---

[1] Allerdings stecken in der Konzeption einer interaktiven Problemlösung wohl stets Erwartungen über die Denkprozesse eines Benutzers; vgl. hinten Überlegungen zur Einbeziehung eines Benutzers in ein Modell für interaktive Problemlösungen, (p.69/70).

(iii) 2. der Benutzer hat die Möglichkeit, unabhängig von dem Zustand, in dem sich ein Problemlösungsprozeß befindet, in diesen einzugreifen; solche Eingriffe werden in (KUPKA-WILSING,1974) "starke Unterbrechungen" genannt. Beispiele dafür sind etwa Anhalten einer von einem Rechner auf einem Datensichtgerät sichtbar gemachten noch nicht abgeschlossenen Nachricht und Benutzung von nicht in einer Problemlösung vorgesehenen Rechnerfunktionen.

(iv) Benutzeraktivitäten im Kommunikationsprozeß sind dadurch gekennzeichnet, daß sie durch Wahl eines Benutzers aus einer Menge von Möglichkeiten hervorgehen; diese Wahl ist im allgemeinen a priori nicht deterministisch festgelegt[1]. Ein Benutzer wirkt damit auf zwei unterscheidbare Weisen, die sich nicht ausschließen, auf einen Lösungsprozeß ein[2]:

---

[1] Vgl.(KUPKA-WILSING,1975), p.40: "Als Grundprinzip des Dialogs zwischen Mensch und Rechnersystem gilt, daß der Benutzer seine Eingaben frei wählt und das System darauf deterministisch reagiert". Diese Betrachtungsweise schließt nicht aus, daß man hoffen kann, daß ein Benutzer in einem Lösungsprozeß ein gewisses "rationales" Verhalten zeigt, auch wenn ihm dies kaum explizit vorgeschrieben werden kann.

[2] Es ist dabei hier unwesentlich, ob das Erbringen von Aktivitäten durch "reflexartiges Reagieren" oder (innovatives) Agieren eines Benutzers ((CARBONELL,1969), p.17) in einem "Sklaven- oder Herrensystem" ((MERTENS-KRESS,1970), (LOSBICHLER-MÜHLBACHER,1974)) erfolgt, oder etwa ein "balanced man-machine system" ((PRESS,1971),p.63) vorliegt.

(iv)1. durch Eingabe von Daten, die die Folge der in einem Lösungsprozeß durchzuführenden Lösungsschritte beeinflußt (vgl. 2.1.2.(iv)1.);

(iv)2. durch Eingabe von Daten, die in die Berechnung von Werten für Ausgabevariable einfließen (vgl. 2.1.2. (iv)2.);

Im ersten Fall wird also in den Kontrollfluß, im zweiten in den Datenfluß[1] einer interaktiven Problemlösung eingegriffen; beide Einwirkmöglichkeiten können auch an einer Unterbrechung kombiniert auftreten.

(v) Rechneraktivitäten im Kommunikationsprozeß bestimmen sich deterministisch[2] aus den implementierten Problemlösungskomponenten und den Daten, die diese zur Verfügung haben, wenn man von möglichem, "nichtdeterministischem" Fehlverhalten realer programmierter Rechner absieht.

Neben den in der Regel beobachtbaren Beschreibungsmerkmalen (i) - (v) werden nun noch zwei Forderungen aufgestellt, denen die hier betrachteten interaktiven Problemlösungen genügen sollen:

(vi) mögliche Lösungsprozesse sind sequentiell in dem Sinne, daß zu einem Zeitpunkt höchstens genau eine "Operation"vom Rechner oder Benutzer durchgeführt wird[3]; eine Operation eines Benutzers sei dabei eine von ihm durchgeführte abgeschlossene Eingabe von Daten.

---

[1] Zum Kontroll- und Datenfluß in einer interaktiven Problemlösung vgl. (SCHMID,1974), p.102ff.

[2] Vgl. Fußnote 1 auf p.55.

[3] Zu diesem Begriff eines sequentiellen Prozesses vgl. (BRINCH HANSEN,1973), p.29.

Diese Forderung schließt zunächst[1] eine Betrachtung von interaktiven Problemlösungsprozessen aus, die parallele Prozesse enthalten[2].

Schließlich sollen die interaktiven Problemlösungen noch der (von der Anwendung her gesehen sinnvollen) Forderung genügen:

(vii) einem Benutzer stehen an jeder Stelle einer schwachen Unterbrechung endlich viele Aktivitäten (d.h. Möglichkeit einer Wahl aus einer endlichen Menge von Daten) zur Verfügung, von denen jede eine eindeutige Fortsetzung eines an dieser Stelle angekommenen Lösungsprozesses bewirkt.

Starke Unterbrechungen (vgl. (iii)2.) werden in einer interaktiven Problemlösung in der Regel nicht explizit vorgesehen und sind damit für deren Konzeption und "Modellierung" kaum von Interesse. Sie sollen daher im folgenden nicht weiter mitbetrachtet werden - was jedoch, entsprechend des Charakters starker Unterbrechungen, deren Verwendung in Lösungsprozessen nicht ausschließt.

---

[1] Die Betrachtungen in 3.3.2.2. erlauben es, Forderung (vi) - bezogen auf eine "äquivalente" Formulierung des Modells interaktive Prozedur - abzuschwächen;

[2] Für einen Beschreibungsansatz, der interaktive Problemlösungsprozesse als parallele Prozesse von Rechner und Benutzer betrachtet, siehe (GRIESE-MÜHLBACHER,1976).

Mit den Merkmalen (i) - (vii), abzüglich (iii)2., einer interaktiven Problemlösung liegt nun die Beschreibung eines "explizit ausgekoppelten Totalitätsauschnitts[1)]" vor, zu dem ein formales Modell erstellt werden kann. Es ist dabei zu beachten, (vgl. auch (WINAND,1975)) daß das Aufstellen der Punkte (i) - (vii) eine unter dem Gesichtspunkt "problembezogene Modellbildung für interaktive Problemlösungen" subjektive Auswahl (Eingrenzung) von Beschreibungsmerkmalen realer interaktiver Problemlösungen beinhaltet.

---

1) Vgl. (WINAND,1975), p.207.

2.3. Interaktive Prozeduren

2.3.1. Formale Definition interaktiver Prozeduren

Es bezeichne Q einer nichtleere endliche Menge von Zuständen ("Problemlösungsstadien"). In Q sei eine Teilmenge Q_U von Unterbrechungszuständen ("Stellen schwacher Unterbrechungen", vgl.2.2.(iii)1.) ausgezeichnet; weiter sei in Q ein Anfangszustand q_o ("Starten eines Problemlösungsprozesses", vgl. 2.2.(iii)1.) ausgezeichnet. A sei ein nichtleeres Alphabet ("geeignetes Alphabet" aus 2.2.(i)); A^* bezeichne die Menge aller Worte über A; es sei $A^+ := A^* - \{\varepsilon\}$, ε Leerwort in A^*. Es sei {l,r,s} eine Menge, die mit $A \cup Q$ elementfremd ist. Weiter bezeichne d auf einer nichtleeren Menge $M \subset Q \times A$ eine Abbildung

$$d : M \to P^+(Q \times \{l,r,s\} \times P^+(A^+)) \quad ,$$

wobei P^+ die Potenzmengenbildung nichtleerer Teilmengen bedeute[1].

Definition 2.3.1.1.: Seien Q, Q_U, A, d und q_o mit den oben angegebenen Bedeutungen[2]; ein Tupel $T=(Q,Q_U,A,d,q_o)$ heiße eine interaktive Prozedur[3], wenn die Abbildung

$$d : M_T \to P^+(Q \times \{l,r,s\} \times P^+(A^+)) \quad , \quad M_T \subset Q \times A, \quad M_T \neq \emptyset \quad ,$$

erfüllt:

(i) für alle $(q,a) \in M_T$ ist $|d(q,a)| < \infty$ und $|A'| < \infty$ für alle A', wenn $d(q,a) = \{(q', b', A'), q' \in Q', b' \in B', A' \in \bar{A}\}$, $Q' \subset Q$, $B' \subset \{l,r,s\}$, $\bar{A} \subset P^+(A^+)$ ist ("Endlichkeitsannahme", vgl. 2.2.(vii));

---

[1] Für eine nichtleere Menge X sei $P^+(X) := \{Y : Y \subset X, Y \neq \emptyset\}$.

[2] Die Bezeichnungen werden im folgenden beibehalten.

[3] Der Begriff "interaktive Prozedur" aus Def. 2.3.1.1. ist als eine Weiterentwicklung des gleichnamigen Begriffes aus (ROTH, 1975a,b) anzusehen.

(ii) ist $q \notin Q_U$ oder $q = q_o$, so ist $|d(q,a)| = 1$ und $A' = \{a'\}$ für ein $a' \in A$, bei $d(q,a) = \{(q', b', A')\}$, ("nichtdeterministisches Verhalten nur an vom Start verschiedenen Unterbrechungen", vgl. 2.2.(iv) und 2.2.(v));

(iii) für $d(q,a) = \{(q'_1,b'_1,A'_1),\ldots,(q'_r,b'_r,A'_r)\}$ [1], $q'_i \in Q$, $b'_i \in \{l,r,s\}$, $A'_i \in P^+(A^+)$ $i=1,..,r$, $r \in \mathbb{N}$ [2], sind alle Wortmengen A'_i paarweise disjunkt ("Eindeutigkeitsforderung", vgl. 2.2.(vii)).

Die Eindeutigkeitsforderung aus Def. 2.3.1.1.(iii) ist wegen Def. 2.3.1.1.(ii) für alle Nichtunterbrechungszustände trivial erfüllt. Bevor der Begriff "interaktive Prozedur" nun näher betrachtet wird, wird zunächst noch definiert, auf welche Weise innerhalb einer interaktiven Prozedur "Arbeitsmöglichkeiten" bestehen sollen. Wir vereinbaren zusätzlich, daß die Zeichen aller Worte w aus A^+ stets von 1 ab "von links her" durchnummeriert sind, also $w = w_1 \ldots w_r$, $r \geq 1$ ist. Bezeichnet man für eine interaktive Prozedur $T = (Q,Q_U,A,d,q_o)$ ein Tupel $(q,w,a(w))$ mit $q \in Q$, $w \in A^+$, $a(w) \in \mathbb{N}$ ("aktuelles Zeichen[3] in w") als eine Konfiguration ("Problemlösungssituation") der interaktiven Prozedur T, so läßt sich damit in (automatentheoretisch) bekannter Weise erklären, wie eine Konfiguration aus einer anderen ableitbar ist:

---

[1] Aufgrund von (i) hat $d(q,a)$ für alle $(q,a) \in M_T$ eine Darstellung der angegebenen Art.

[2] $\mathbb{N}$ bezeichne die natürlichen Zahlen ausschließlich Null.

[3] Zwischen dem Index eines aktuellen Zeichens und dem aktuellen Zeichen aus A selbst wird im folgenden nicht unterschieden, wenn keine Unklarheiten zu befürchten sind.

Definition 2.3.1.2.: Sei $T = (Q,Q_U,A,d,q_o)$ eine interaktive Prozedur; eine Konfiguration $(q', w', a(w'))$ von T heiße aus einer Konfiguration $(q,w,a(w))$ von T über T ableitbar, wenn gilt:

(i) $\exists\, w'' \in A^+$ und $\exists\, A' \in P^+(A^+)$ mit $w'' \in A'$ derart, daß $(q', b', A') \in d(q,a(w))$ ist, $b' \in \{l,r,s\}$;

(ii) ist $w = w_1 \ldots w_m$, so ist

$w' = w_1 \ldots w_{i_o-1} w'' w_{i_o-1} \ldots w_m$ [1], falls $a(w) = w_{i_o}$ ist, und mit $w'' = w_1'' \ldots w_s''$ $a(w')$ wie folgt definierbar ist:

$$a(w') = \begin{cases} w_{i_o-1}, & \text{falls } b' = l \text{ und } i_o > 1, \\ w_1'', & \text{falls } b' = l \text{ und } i_o = 1, \\ w_{i_o}, & \text{falls } b' = s \text{ und } w'' \text{ ein Zeichen aus A ist,} \\ w_{i_o+1}, & \text{falls } b' = r \text{ und } i_o < m, \\ w_s'', & \text{falls } b' = r \text{ und } i_o = m. \end{cases}$$ [2]

"Arbeitsmöglichkeiten" innerhalb einer interaktiven Prozedur werden dann wie folgt definiert:

Definition 2.3.1.3.: Sei $T = (Q,Q_U,A,d,q_o)$ eine interaktive Prozedur; eine Folge $t = \{(q^{(i)},w^{(i)},a(w^{(i)}))\}_{i=0,1,..}$ von Konfigurationen von T ("sequentielle Lösungsprozesse", vgl. Forderung 2.2.(vi)) heiße ein innerhalb T durchführbarer Lösungsprozeß (kurz: Lösungsprozeß in T), wenn gilt:

---

[1] Man beachte, daß sich die Inzidierung von w' noch auf die Indizierung von w bezieht.

[2] a(w') ist nicht für alle Einsetzungsmöglichkeiten von w'' in w definiert.

(i) $q^{(0)} = q_0$, $a(w^{(0)}) = w_{i_0}^{(0)}$, $w_{i_0}^{(0)}$ Zeichen aus w,

(ii) $(q^{(i+1)}, w^{(i+1)}, a(w^{(i+1)}))$ ist über T aus $(q^{(i)}, w^{(i)}, a(w^{(i)}))$ für alle i = 0,1,.. ableitbar.

t heiße eine Interaktion, wenn zusätzlich gilt:

(iii) t ist endlich, $t = \{(q^{(i)}, w^{(i)}, a(w^{(i)}))\}_{i=0,..,n}$, $n \geq 1$, und es existiert keine aus $(q^{(n)}, w^{(n)}, a(w^{(n)}))$ ableitbare Konfiguration von T.

Aus den bereits in Klammern angefügten kurzen Anmerkungen zu den Bezügen der Definitionen 2.3.1.1. und 2.3.1.3., sowie der weiteren eingeführten Begriffe zu Teil 2.2. geht bereits grob der Zusammenhang der einzelnen formalen Begriffe zu den Beschreibungsmerkmalen für interaktive Problemlösungen hervor. Um eine genauere Verbindung des hier entwickelten Modelles "interaktive Prozedur" und der in ihm möglichen Lösungsprozesse zu den allgemeinen Beschreibungsmerkmalen aus 2.2. herzustellen - schließlich sollen interaktive Prozeduren interaktive Problemlösungen "möglichst gut" [1] darstellen - , ist jedoch noch eine ausführliche Betrachtung notwendig.

Eine interaktive Problemlösung gliedert sich in einem Rechner und einem Benutzer übertragene Lösungskomponenten auf. Da hier nach Voraussetzung digitale reale Rechner zugrunde liegen, die (in endlicher Zeit) mit Hilfe endlich vieler Zustände charakterisierbar sind, ist es sinnvoll, für die Beschreibung der einem Rechner übertragenen Lösungskomponenten endlich viele Zustände zugrunde zu legen.

---

[1] Vgl. dazu die Homomorphie- und Isomorphiebetrachtungen etwa in (KIRSCH,1971),p.34ff. und (DINKELBACH,1973).

Von einem Benutzer wird im Rahmen einer interaktiven Problemlösung erwartet, daß er stets bereit ist, Aktionen innerhalb des Lösungsfortganges dann durchzuführen, wenn ihm dies an Unterbrechungen möglich ist. Nimmt man dementsprechend für einen Benutzer - global betrachtet - stets den Zustand "bei Unterbrechungen agierend" an, so ist es möglich, das statische Aussehen einer interaktiven Problemlösung durch endlich viele Zustände zu beschreiben. Hier wird dies durch die endliche Zustandsmenge Q und eine darin ausgezeichnete Teilmenge von Unterbrechungszuständen Q_U dargestellt; die Unterbrechungszustände ermöglichen dabei einem Benutzer ein Aktivwerden[1)]. Ob er allerdings bei einer Unterbrechung aktiv wird, ist für eine statische Lösungsbeschreibung zunächst ohne Bedeutung, da diese nur die Möglichkeiten zum Aktivwerden angibt; erst beim Betrachten von Lösungsprozessen ist Aktivwerden eines Benutzers wesentlich, vgl. den Existenzoperator "$\exists$" in Def. 2.3.1.2.(i). Ein Zustand einer interaktiven Problemlösung, in dem Problemlösungsprozesse begonnen werden können, wird durch den Anfangszustand q_o im Modell ausgezeichnet; da dieses Starten von einem Benutzer ausgehen kann, kann q_o ein Unterbrechungszustand sein.

[1)] Vgl. die Annahme über den Zustand eines Benutzers "bei Unterbrechungen agierend".

Bei einer Unterbrechung und nur dann stehen einem Benutzer endlich viele Möglichkeiten zur weiteren Beeinflussung eines Lösungsprozesses zur Verfügung (vgl. 2.2.(vii) sowie 2.2.(ii)1. u. 2.); in der Definition 2.3.1.1. des Modells "interaktive Prozedur" wird dies durch die Gestalt der Abbildung d als Abbildung in $P^+(Q\times\{l,r,s\} \times P^+(A^+))$ zusammen mit der Forderung (ii) dargestellt. Ist etwa $d(q,a) = \{(q'_1,b'_1,A'_1),..,(q'_r,b'_r,A'_r)\}$ für ein $(q,a) \in M_T$, $q \in Q_U$, so stellen die Wortmengen A'_i jeweils den Spielraum dar, den ein Benutzer an der Unterbrechungsstelle q zur Verfügung hat, wenn er einen Lösungsprozeß an der Stelle q'_i fortsetzen will; $q'_1,...,q'_r$ sind die Stellen in der Problemlösung, die ein Benutzer an der Stelle q als Spielraum zur Fortsetzung eines Lösungsprozesses zur Verfügung hat; durch die Wahl eines Wortes aus $A'_1 \cup ... \cup A'_r$ legt ein Benutzer eine Stelle aus $\{q'_1,...,q'_r\}$ eindeutig (vgl. Def. 2.3.1.1.(iii)) fest. Außerhalb der Unterbrechungszustände wird in Analogie zu einem realen digitalen Rechner zeichenweise gearbeitet. Die Endlichkeitsannahme aus 2.2.(vii) an die Aktivitäten eines Benutzers findet in Forderung (i) aus Def. 2.3.1.1. ihren Ausdruck. Die Eindeutigkeitsforderung an Benutzereingriffe aus 2.2.(vii) wird von Def. 2.3.1.1.(iii) im Modell gegeben Das außerhalb der Aktivitätsmöglichkeiten eines Benutzers geforderte deterministische Verhalten interaktiver Problemlösungen,(vgl. 2.2.(v)), wird von Forderung (ii) in Def. 2.3.1.1. berücksichtigt; sie bedeutet insbesondere, daß die Konfigurationen eines Lösungsprozesses "außerhalb" von Unterbrechungszuständen in eindeutig bestimmter Weise auseinander hervorgehen.

Interaktive Prozeduren sind also ein Modell, das in enger Weise interaktive Problemlösungen, die den Beschreibungsmerkmalen aus 2.2. genügen (abzüglich 2.2.(iii)2.) formal darstellt, wenn man akzeptiert, daß sich das mögliche Agieren eines Benutzers formal als Auswahl aus gewissen (Wort)Mengen wiedergeben läßt[1]. Zugleich sind interaktive Prozeduren ein Begriff, der Rechner und Benutzer in einer interaktiven Problemlösung auf einer Ebene, etwa im Sinne der von Licklider geforderten "man-computer-symbiosis"[2] in ein Modell einbezieht.

(Sequentielle) Lösungsprozesse innerhalb einer interaktiven Problemlösung werden dann durch Folgen von auseinander ableitbaren Konfigurationen dargestellt, deren Indexmengen Folgen von Zeitpunkten modellhaft beschreiben. Eine in einem Lösungsprozeß erreichte Stelle in einer Problemlösung, sowie die bis dahin erarbeitete Information, stehen dabei in den ersten beiden Stellen einer Konfiguration (q,w,a(w)); die letzte - a(w) - weist auf die an dieser Stelle besonders "aktuelle" Information hin. Modelle von Lösungsprozessen endlicher Länge, die keine Fortsetzung haben, sollen (vgl. Def. 2.3.1.3.) als Interaktionen bezeichnet werden.

---

[1] Vgl. 2.2.(iv).

[2] Vgl. (LICKLIDER,1960).

In der Literatur sind - gemessen an der Bedeutung von interaktiven Problemlösungen - relativ wenige Ansätze zu ihrer formalen Behandlung bekannt. Es lassen sich folgende Beschreibungsschwerpunkte finden:

a) Das Verhalten von Benutzer und Rechner wird durch einen geeigneten Automaten dargestellt. In (KUPKA 1972), (KUPKA-WILSING,1972,1973,1974,1975) werden dazu unendliche Mealy-Maschinen[1] benutzt; in einem Zustand ist eine Eingabe - ein Wort über einem Eingabealphabet - als Aktivität eines Benutzers und die resultierende Ausgabe - ein Wort über einem Ausgabealphabet - als die darauf erfolgende Reaktion eines Rechners auffaßbar; Ziel der Untersuchungen ist es im wesentlichen, allgemeine Aussagen über Syntax und Semantik von Dialogsprachen[2] zu gewinnen. Auf dem Übergangsverhalten von sequentiellen Maschinen - veranschaulicht durch sog. Transitionsdiagramme[3] - fußen die Problemlösungsbeschreibungen etwa in (HEISTAND,1964) (interaktive Komponenten eines Betriebssystems), (HÄNDLER,1968) (Betriebssystem), (NEWMAN,1968) (interaktive Lösung graphischer Probleme),(BROWN-BURTON-ZDYBEL,1973)(Rechner-gestützter Meteorologieunterricht), (LEBEUX,1973) (Syntaxgesteuerte Programmierung),

---

1) Zur Definition einer Mealy-Maschine vgl. (BOOTH,1967), p.69ff.; hier kommt hinzu, daß unendlich viele Eingaben und Ausgaben zugelassen sind.

2) Zum Begriff Dialogsprache, insbesondere im Zusammenhang mit Programmiersprachen wie FORTRAN, ALGOL, u.ä. siehe (KUPKA-WILSING,1975), p.48ff.

3) Vgl. (BOOTH,1967), p.72ff.

(DENERT,1975) (interaktives Problemlösen mit dem "PLAN2D-Editor", dargestellt durch "hierarchische endliche Automaten"); in allgemeinem Rahmen wird die Benutzung von Transitionsdiagrammen bei der Beschreibung interaktiver Problemlösungen in (PARNAS,1969) diskutiert[1].

Entsprechend den Eigenschaften einer sequentiellen Maschine sind derartige Beschreibungen für interaktive Problemlösungen sequentiell Input-Output-orientiert; das bedeutet hier: dialogorientiert[2].

b) Der Beschreibungsschwerpunkt liegt auf der Kommunikation zwischen Benutzer und Rechner; formales Hilfsmittel zur Beschreibung von "Dialogsystemen" sind auf Petri zurückgehende[3] Netze aus Instanzen und Kanälen; entsprechende Modelle finden sich etwa in (SCHMID,1974), insbesondere in Teil 5., (HÖNS,1974), insbesondere in Teil 1. sowie (OBERQUELLE,1975) (sehr allgemein). Bei derartigen Beschreibungen kann ein Benutzer über "makroskopische" Zustände, wie "Benutzer wartet", "Benutzer erhält die Kontrolle", u.ä. einbezogen werden[4].

---

[1] Vgl. auch (JOHNSON,1968),p.161.

[2] Ein Dialog ist eine abwechselnde Kommunikation von Rechner und Benutzer, vgl. (KUPKA-WILSING,1975),p.42; zur formalen Darstellung vgl. 2.2.(i), sowie im Rahmen des Modells "interaktive Prozedur" vgl. 3.1.1.

[3] Vgl. (PETRI,1962).

[4] Vgl. (SCHMID,1974), p. 106ff.; in allgemeinerem Rahmen vgl. zur Darstellung eines Benutzers in diskreten Zuständen (ANGEL-BEKEY,1968).

In Verbindung mit dem in a) beschriebenen automatentheoretischen Beschreibungsansatz können die Übergänge in solchen Netzen spezifiziert werden, vgl. (KUPKA-WILSING,1975), p.134ff.

c) Die Folge der Aktionen eines Rechners und eines Benutzers werden als (zwei) parallele Prozesse beschrieben, die "geeignet" synchronisiert[1] werden müssen; ein derartiger Beschreibungsansatz findet sich in (GRIESE-MÜHLBACHER, 1976). Synchronisationsprobleme - speziell für im Rechner ablaufende Prozesse - sind allerdings generell bei der Konzipierung und Implementierung von (interaktiven) Problemlösungen zu berücksichtigen, was etwa in (DENERT,1975) bei der Implementierung des interaktiven sog. "PLAN2D-Editors"[2] deutlich wird[3].

Im Vergleich zu den geschilderten Beschreibungsmethoden[4]

---

[1] Zur Lösung solcher Synchronisationsprobleme vgl. (BRINCH HANSEN,1973), Teil 3.

[2] Zu dessen Funktion siehe (DENERT,1975).

[3] Vgl. (DENERT,1975), insbesondere p. 13, 14 sowie p.36.

[4] Nicht unerwähnt bleiben sollen in diesen Zusammenhang an "makroskopischen" Rechnerzuständen orientierte Beschreibungen von implementierten Programmsystemen, im wesentlichen Betriebssystemen, mit deren Hilfen Aussagen über ihr "physikalisches Verhalten"in interaktiven Lösungsprozessen, wie"Ressourcenverteilung", durchschnittliche Antwortzeiten, u.ä. gewonnen werden können, vgl. etwa (NICKERSON,1969), Teil 4, (HAAG,1968) (sehr allgemein); sie sind allerdings keine Modelle für interaktive Problemlösungen, wie sie hier betrachtet werden.

für interaktive Problemlösungen ist das Modell "interaktive Prozedur" aus Def. 2.3.1.1. "implementierungsnäher", da es nicht nur die Reaktion eines Rechners auf eine Benutzeraktivität enthält, sondern auch wie diese Reaktion aufgrund eines Turingprogrammes zustande kommt. Dadurch wird es möglich, auf der Basis eines Modelles Aussagen über implementierte interaktive Problemlösungen zu gewinnen, wie etwa über ihre Korrektheit[1] oder ihre Komplexität[2] - unter Einbeziehung der Aktivitäten eines Benutzers -, was mit den oben beschriebenen Modellen für interaktive Problemlösungen kaum möglich ist.

In allen Modellen, einschließlich des hier entwickelten, erscheint ein Benutzer im wesentlichen als eine "black box", die höchstens durch "makroskopische"[3] Benutzerzustände etwas spezifiziert wird. Das Verhalten eines Benutzers wird dabei durch freies Auswählen[4] im Rahmen von gewissen Möglichkeiten modellhaft dargestellt[5].

---

[1] Vgl. 3.4.

[2] Vgl. 3.5.

[3] Vgl. im Modell "interaktive Prozedur" den Benutzerzustand "bei Unterbrechungen agierend", sowie in (SCHMID,1974), p.106ff.

[4] Vgl.(KUPKA-WILSING,1975), p.120 :"Wir setzen (dabei) voraus, daß der Benutzer seine Eingaben frei wählen kann"; (SCHMID, 1974), p.103 :"In diesem Zustand (der Benutzer hat die Kontrolle über ein Dialogsystem, d.Verf.) ist es in seinem (des Benutzers,d.Verf.) Belieben, wann die nächste Eingabe stattfinden und woraus sie bestehen wird."

[5] Dies schließt nicht aus, daß man in vielen Lösungsprozessen in interaktiven Problemlösungen ein rationales, zielorientiertes Verhalten eines Benutzers erwartet oder zumindest erhofft, vgl. 2.2.(iv).

Demgegenüber stehen Modelle für interaktives Problemlösen, die das Verhalten eines Benutzers ebenfalls ausführlich in formalen (Teil-)Modellen einbeziehen wollen, vgl. etwa (CARBONELL,1969), (KIESER,1973); es wird dabei unterstellt, daß sich menschliche Denkstrukturen im Rahmen eines informationsverarbeitenden Systems[1] modellhaft darstellen lassen[2]. Da solche Modelle aber bislang Benutzerverhalten nicht so beschreiben können[3], daß funktional das Entstehen von Benutzeraktivitäten aus Rechneraktivitäten sowie weiteren Informationen deutlich wird[4], erscheint die obige Annahme über das Benutzerverhalten - freies Auswählen - gerechtfertigt.

---

[1] In (KIESER,1973), p.44 als "individuelles Informationsverarbeitungssystem" bezeichnet.

[2] Vgl. (NEWELL-SIMON,1972), p.788ff.; (KIESER,1973),p.43ff.

[3] Dahinter steht die Frage, wie genau sich menschliches Denkverhalten überhaupt in Modelle fassen läßt; darauf soll hier jedoch nicht weiter eingegangen werden.

[4] Vgl. etwa in (CARBONELL,1969) das Benutzer-Rechner-Modell in Fig. 3 auf p. 19, in dem der Benutzer aus der (nicht spezifizierten) "lösungsstrategieorientierten Verarbeitung" von Rechneraktivitäten und Aktivitäten, die sein "internal model" (ebenfalls nicht spezifiziert) liefert, seine nächste Aktivität in einem interaktiven Lösungsprozeß bestimmt.

2.3.2. Interaktive Prozeduren und Turingprogramme

Es soll nun ein Zusammenhang zwischen interaktiven Prozeduren und Turingmaschinen[1], auf den im folgenden verschiedentlich zurückgegriffen wird, aufgezeigt werden. Dazu sind einige Vorbetrachtungen erforderlich.

Sei $T = (Q,Q_U,A,d,q_o)$ eine interaktive Prozedur; dann kann man d in üblicher Weise durch ein Zustandsdiagramm (Transitionsdiagramm) - einen endlichen (gerichteten) bewerteten Graphen[2] G(T) ohne Parallelen - darstellen: G(T) habe die Eckenmenge Q (Zustandsmenge von T); seine Pfeilmenge R sei eine Teilmenge aus $Q \times Q$; ein Pfeil mit Anfangsecke q und Endecke q' gehöre zu R genau dann, wenn ein $a \in A$ existiert, so daß d(q,a) definiert ist und folgendes erfüllt: ist $d(q,a) = \{(q'_1,b'_1,A'_1),..,(q'_r,b'_r,A'_r)\}$, so ist $q' = q'_{i_o}$ für ein $i_o \in \{1,..,r\}$. Ein Pfeil aus R werde dabei mit der Menge aller durch d gegebenen Möglichkeiten und ihren Resultaten bewertet, mit denen in T ein Übergang vom Zustand q in den Zustand q' möglich ist, wenn q Anfangs- und q' Endecke dieses Pfeils ist;

---

[1] Zum Begriff siehe etwa (HOPCROFT-ULLMAN,1969), chap.6, (BRAUER-INDERMARK,1968), §2; später wird im wesentlichen der Begriff "Turingprogramm" nach (BRAUER-INDERMARK,1968), §3, benutzt.

[2] Für die graphentheoretischen Begriffe siehe (NOLTEMEIER, 1976), Teil 1.2.

genauer:

$c : R \longrightarrow P(\{(a,w,b),\ a \in A,\ w \in A^+,\ b \in \{l,r,s\}\})$,

definiert durch

$c(q,q') := \{(a,w,b) : (a,q) \in d^{-1}(\{(q',b,A')\})$[1)], $A' \subset A^+$ und $w \in A',\ b \in \{l,r,s\}\}$;

Es ist klar, daß man auch mittels eines Graphen der beschriebenen Art eine Abbildung d einer interaktiven Prozedur definieren kann.

Für ein Zustandspaar $(q,q') \in ((Q - Q_U) \cup \{q_o\}) \times Q$ bezeichne $W(q,q')$ die Menge aller Zustände von T, die auf der Spur eines Weges[2)] in G(T) von q nach q' liegen, der keinen Unterbrechungszustand außer höchstens q' oder q_o enthält.

Für ein $q \in (Q - Q_U) \cup \{q_o\}$ und eine Menge $Q' \subset Q$, $q \in Q'$ sei

$B_{Q'}(q) := \{(q,a,a',b,q') : a,a' \in A,\ b \in \{l,r,s\},\ q' \in Q'$ und $d(q,a) = \{(q',b,\{a'\})\}$[3)]$\}$.

$B_{Q'}(q)$ gibt alle durch d bestimmten Möglichkeiten an, von einem Nichtunterbrechungszustand q aus einen weiteren (ggf. den gleichen) Zustand aus Q' von T zu erreichen.

Dann gilt:

[1)] $d^{-1}(\{(q',b,A')\})$ bezeichne die Urbildmenge zu (q',b,A') unter d.

[2)] Zur Definition der Begriffe Weg und Spur siehe wieder (NOLTEMEIER,1976), p.28.

[3)] Da $q \in (Q - Q_U) \cup \{q_o\}$ sein soll, hat d(q,a) das angegebene einfache Aussehen, vgl. Def. 2.3.1.1.(ii).

Satz 2.3.2.1.: Seien $T = (Q,Q_U,A,d,q_o)$ eine interaktive Prozedur und $(q,q') \in ((Q - Q_U) \cup \{q_o\}) \times Q$ ein Zustandspaar derart, daß $W(q,q')$ nichtleer ist, $W(q,q') = \{q_1:=q,..,q_r=:q'\}$[1]; dann gilt:

(i) $P_T := \{B_{W(q,q')}(q_1),...,B_{W(q,q')}(q_{r-1}), \bar{B}(q_r)\}$ mit

$$\bar{B}(q_r) := \begin{cases} \emptyset, \text{falls } q' \in Q_U \\ B_{W(q,q')}(q'), \text{falls } q' \notin Q_U \end{cases}$$

ist ein Turingprogramm bezüglich des Alphabetes A und der Zustandsmenge $W(q,q')$;

(ii) sind $t = \{(q^{(i)},w^{(i)},a(w^{(i)}))\}_{i=0,1,..}$ ein Lösungsprozeß in T und q^{i_o}, q^{i_o+1} für ein i_o aus der Indexmenge von t Zustände aus $W(q,q')$, so ist $B_{W(q,q')}(q^{(i_o)})$ ein mit der Information $(w^{(i_o)},a(w^{(i_o)})$ durchführbarer, als Folgeinformation $(w^{(i_o+1)},a(w^{(i_o+1)}))$ ergebender Befehl des Turingprogrammes P_T[2] .

Beweis: (i) Da nach Definition von $W(q,q')$ alle Zustände aus $W(q,q')$ außer höchstens q' oder q_o keine Unterbrechungszustände sind, sind, wegen Definition 2.3.1.1.(ii), die Mengen $B_{W(q,q')}(q_1),...,B_{W(q,q')}(q_{r-1})$ sowie $B_{W(q,q')}(q_r)$, falls $q_r \notin Q_U$, Befehle bezüglich des Alphabets A.

---

[1] Offenbar ist $W(q,q')$ für alle Zustandspaare aus $((Q - Q_U) \cup \{q_o\}) \times Q$ stets leer oder endlich.

[2] Für die Begriffe siehe (BRAUER-INDERMARK,1968), §3; der Index eines Zustandes q_i eines Befehles $B_{Q'}(q_i)$ soll stets auch der Index des Befehles sein, so daß etwa in (i) q_1 Anfangszustand ist (vgl. (BRAUER-INDERMARK, 1968), Def. 3.4.).

Da auch die leere Menge ein Befehl ist, ist insgesamt $\{B_{W(q,q')}(q_1),\ldots,B_{W(q,q')}(q_{r-1}), \bar{B}(q_r)\}$ ein Turingprogramm bezüglich A zur Zustandsmenge W(q,q').

(ii) Da i_o, i_o+1 nach Voraussetzung aus der Indexmenge von t sein sollen, ist aufgrund von Def. 2.3.1.3. $(q^{(i_o+1)},w^{(i_o+1)},a(w^{(i_o+1)}))$ aus $(q^{(i_o)},w^{(i_o)},a(w^{(i_o)}))$ ableitbar. D.h. insbesondere, daß $B_{W(q,q')}(q^{(i_o)})$ einen Befehl $x = \{(q^{(i_o)},a(w^{(i_o)}), a', b', q^{(i_o+1)})\}$ mit a', b' bestimmt aus $d(q^{(i_o)},a(w^{(i_o)})) = \{(q^{(i_o+1)},b',\{a'\})\}$ enthält. $B_{W(q,q')}(q^{(i_o)})$ ist also bei der Information $(w^{(i_o)},a(w^{(i_o)})$ durchführbar. Wegen Def. 2.3.1.2. ist nach Durchführung von x bei der Information $(w^{(i_o)},a(w^{(i_o)}))$ dann offenbar $(w^{(i_o+1)},a(w^{(i_o+1)}))$ die Folgeinformation.

q.e.d.

Korollar 2.3.2.1.: Seien $T = (Q,Q_U,A,d,q_o)$ eine interaktive Prozedur mit $Q_U - \{q_o\} = \emptyset$ und $Q = \{q_o,\ldots,q_n\}$, $n \geq 0$; dann ist $P_T = \{B_Q(q_o),\ldots,B_Q(q_n)\}$ ein Turingprogramm, dessen nichttriviale[1] Anwendungen[2] auf Informationen der Art (w,a(w)) mit $w \in A$, a(w) (aktuelles) Zeichen aus w genau die Interaktionen in T sind.

Der Beweis folgt unmittelbar aus Satz 2.3.2.1.

---

[1] Nichttrivial bedeute Durchführung mindestens eines Befehles aus P_T.

[2] Zum Begriff Anwendung eines Turingprogrammes siehe (BRAUER-INDERMARK,1968), p.18.

Satz 2.3.2.1. und Korollar 2.3.2.1. zeigen eine Struktur von interaktiven Prozeduren auf, die es erlaubt, interaktive Prozeduren über die getroffenen Definitionen hinaus zu charakterisieren und in einen Zusammenhang mit durch Turingmaschinen(-programmen) gegebene Problemlösungen zu setzen. Interaktive Prozeduren setzen sich dementsprechend aus Kommunikationsteilen, in denen ein Benutzer Aktivitäten ergreifen kann und "Rechnerteilen" zusammen, deren Funktion durch Turingprogramme beschrieben werden können[1] (Satz 2.3.2.1.); der genaue Zusammenhang wird in 3.2. betrachtet. Falls höchstens der Anfangszustand einer interaktiven Prozedur T ein Unterbrechungszustand ist, ist T insgesamt ein Turingprogramm (Korollar 2.3.2.1.); d.h., wenn also höchstens der Beginn von Lösungsprozessen in einer Problemlösung vom Benutzer gesetzt wird, alle weiteren Lösungskomponenten aber vom Rechner ausgeführt werden, wird, (ggf. bis auf das Starten) vom Rechner eine Prozedur[2] durchgeführt. Prozeduren erweisen sich damit als ein Spezialfall der hier eingeführten interaktiven Prozeduren. Das hat - vom Modell in die Realität übertragen - insbesondere die Bedeutung, daß durch Programme gegebene Problemlösungen, in denen alle Lösungsprozesse ohne Kommunikation mit einem Benutzer ablaufen, als Spezialfälle interaktiver Problemlösungen angesehen werden können[3].

---

[1] Vgl. (CARBONELL,1969), der in seinem benutzerorientierten Modell für interaktive Lösungsprozesse verlangt, daß der Rechner in etwa wie eine Turingmaschine arbeite.

[2] Zur Definition des Begriffes "Prozedur" siehe (HOPCROFT-ULLMAN,1969),§6.4..

[3] Vgl. hierzu auch die Begriffsbestimmungen in (KUPKA-OBERQUELLE-WILSING,1974),in denen Programme als Spezialfall interaktiver Problemlösungen angesehen werden.

Vor weiteren allgemeinen Betrachtungen sollen nun zunächst die eingeführten Begriffe anhand der interaktiven Transportmitteldisposition aus 2.1. verdeutlicht werden.

2.3.3. Darstellung eines Teils der interaktiven Transportmitteldisposition als interaktive Prozedur

Betrachtet man Definition 2.3.1.1. unter dem Aspekt, mit ihrer Hilfe reale interaktive Problemlösungen darzustellen, so ist sofort klar, daß solche Darstellungen aufwendig ausfallen können. Da erst in 3.2. eine zur Definition 2.3.1.1. in gewissem Sinn äquivalente Darstellung gegeben wird, die u.a. eine zusammengefaßte Beschreibung von dem Rechner übertragenen Lösungskomponenten ermöglicht, soll hier zunächst nur ein relativ einfacher Ausschnitt aus der interaktiven Problemlösung der Transportmitteldisposition aus 2.1.1. herausgegriffen werden.

Betrachtet wird nun der in Figur 2.1.1.1. gestrichelt umfaßte Problemlösungsteil, der nach einer Vergabe von Transportmitteln die Möglichkeit zu weiteren Vergaben steuert; eingeschlossen ist eine Fehlerprüfung der möglichen Benutzeraktivitäten.

Seien dazu

$$Q := \{q_0, q_1, q_2, q_{21}, q_{22}\},\ Q_U := \{q_1\},$$

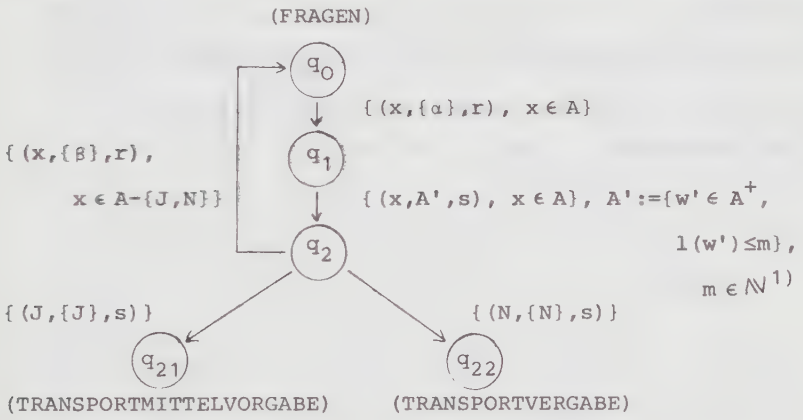
$$A := \{0,1,\ldots,9,A,B,\ldots,Z\} \cup \{\alpha, \beta, _, ,, ?, .\}^{1)},$$

---

1) Das Zeichen _ hat im folgenden stets die Bedeutung einer Leerstelle, etwa vergleichbar mit dem Symbol $\flat$ aus (BRAUER-INDERMARK,1968), Definition 4.2.; man beachte, daß die Satzzeichen Komma, Fragezeichen und Punkt zu A gehören.

wobei α das Literal 'WEITERE_TRANSPORTMITTELVERGABE_GE-WUENSCHT?' und β das Literal 'LETZTE_ANTWORT_FEHLERHAFT,_ DAHER_NICHT_BERUECKSICHTIGT.' bezeichnen. Da im Verlauf der Transportmitteldisposition keine Teile dieser Literale benötigt werden, können sie in der folgenden formalen Darstellung ohne Einschränkung als ein Zeichen angesehen werden.

Mit den gegebenen Mengen Q, Q_U, A und $\{q_0\}$ sei dann $T := (Q, Q_U, A, d, q_0)$, wobei d durch den folgenden Transitionsgraphen G(T) definiert werde:



Figur 2.3.3.1.

Die bei q_0, q_{21} und q_{22} angefügten Namen bezeichnen die Marken zu dem in Figur 2.3.3.1. dargestellten Transportmitteldispositionsteil des in Anhang 1 dokumentierten Programmes.

---

1) l(w') bezeichne die Wortlänge von w'; die Vorgabe einer natürlichen Zahl m zur Längenbeschränkung aller von einem Benutzer ausführbaren Aktivitäten bedeutet für hinreichend großes m keine Einschränkung der Beschreibung.

Es ist leicht ersichtlich, daß T den hier betrachteten Teil der interaktiven Transportmitteldisposition aus 2.1. - gesehen unter dem Gesichtspunkt der Beschreibungsmerkmale aus 2.2.(i)-(vii) - darstellt, wenn man etwa von einem Startwort $w^{(0)} := _..._$, $l(w^{(0)})$ hinreichend groß ("leerer Bildschirm"), ausgeht[1]. Die zwischen Figur 2.3.3.1. bestehende gewisse bildliche Übereinstimmung mit dem gestrichelt umfaßten Problemlösungsteil aus Figur 2.1.1.1. findet in allgemeinem Rahmen ihre Erklärung in 3.2. und 3.3..

$\{B_W(q_2), \overline{B}_W(q_{22})\} = \{\{(q_2,N,N,s,q_{22})\}, \emptyset\}$ mit $W = \{q_2,q_{22}\}$ ist ein Turingprogramm der in Satz 2.3.2.1. betrachteten Art; Lösungsprozesse in T werden in 3.1.2. zusammen mit Dialogen angegeben.

---

[1] Dies stellt bei lokaler Betrachtung der obigen Problemlösungskomponente keine Einschränkung dar; bei globaler Betrachtung hingegen muß $w^{(0)}$ mindestens die Vorgeschichte eines q_0 erreichenden Lösungsprozesses enthalten.

2.4. Zum zeitlichen Charakter interaktiver Problemlösungen

In der Aufstellung von allgemeinen Beschreibungsmerkmalen interaktiver Problemlösungen in 2.2. ist explizit der Faktor Zeit nicht enthalten, wohl aber implizit in der Forderung von in der Zeit sequentiell ablaufenden Lösungsprozessen, vgl. 2.2.(vi). Für die Zeitdauer von Lösungsschritten interaktiver Problemlösungen kann man im engeren Sinne fordern, daß

(i) die verstreichende Zeit von einer abgeschlossenen Benutzeraktivität bis zum Vorliegen einer vollständigen Antwort darauf, die sogenannte Antwortzeit[1], vernachlässigbar kurz[2] ist;

im weiteren Sinne wird man nur fordern, daß

(ii) die einem Rechner übertragenen Problemlösungskomponenten von diesem mit von einem Benutzer nicht als einen Lösungsprozeß hemmend empfundener Antwortzeit durchgeführt werden.

Daneben steht die Antwortzeit eines Benutzers, die sich aus seiner "Denkzeit" nach Abschluß der letzten Aktivität eines Rechners und der Zeit zusammensetzt, die vergeht, bis ein Benutzer eine Aktivität an einer Unterbrechung abgeschlossen

---

[1] Zum Begriff vgl. (DIN,1972); für ausführliche Überlegungen zur Antwortzeit, insbesondere unter psychologischen Aspekten siehe (MARTIN,1973), chap.18; vgl. auch oben Betrachtungen in 1.3..

[2] Im Regelfall setzt dies direktes Verbundensein ("on-line") von Benutzer und Rechner voraus, was häufig als eine Voraussetzung für Problemlösen in Kommunikation zwischen einem Benutzer und einem Rechner verlangt wird, vgl. etwa (MERTENS-KRESS,1970), p.2, (KIESER,1973), p.32.

hat; diese Antwortzeit beeinflußt die Auslastung einer realen Rechenanlage[1].

Als "nicht hemmend" kann nun von einem Benutzer auch die Antwortzeit eines Rechners empfunden werden, die vergeht, bis der Rechner im Stapelbetrieb ein Programm durchgeführt hat ("batch response time"[2]). Auf diese Weise erhält man explizit auch von der Zeit her das Ausführen von Programmen ohne - außer ggf. beim Starten - Benutzereinwirkung als Spezialfall interaktiven Problemlösens[3]. Im weitesten Sinne ist dann auch eine Folge von Ausführungen eines Programmes durch einen Benutzer - etwa das Testen eines Programmes mit jeweils durchgeführten Fehlerkorrekturen mit einem im Stapelbetrieb arbeitenden Rechner - ein auf eine Lösung (möglichst fehlerfreies Programm) hin zielgerichteter Kommunikationsprozeß und damit ein interaktiver Problemlösungsprozeß[4] ("off-line-Kommunikation"). Es ist dabei jedoch zu sehen, daß sich diese Art der Kommunikation und "on-line-Kommunikation" nicht nur zeitlich, sondern auch durch in der Regel

---

[1] Vgl. (BROWN-KLERER,1975); dort wird an einem Warteschlangenmodell der Einfluß der Antwortzeit eines Benutzers auf die Auslastung einer realen, im Time-Sharing-Betrieb arbeitenden Rechenanlage untersucht.

[2] Zum Begriff siehe (BRINCH HANSEN,1973), p.9; der Begriff Antwortzeit nach (DIN,1972) schließt die "batch response time" ein, da er an keine Betriebsart eines Rechners gebunden ist.

[3] Vgl. die Begriffsbestimmungen in (KUPKA-OBERQUELLE-WILSING, 1974) für interaktives Problemlösen sowie Korollar 2.3.2.1. für das hier betrachtete Modell.

[4] Vgl. 1.1.

unterschiedliches Problemlösungsverhalten von Benutzern unterscheiden[1].

Beide Begriffe aus (i) und (ii), "vernachlässigbar kurze Zeit" sowie "nicht als hemmend empfundene Zeit" spiegeln wider, daß es in praxi sehr problematisch ist, Zeiten oder selbst nur Zeitintervalle für die Größe von Antwortzeiten anzugeben; diese hängen zudem stark von den Gegebenheiten (Auslastung, u.ä.) eines realen Rechners ab, mit dem Lösungsprozesse durchgeführt werden.

Die angestellten Überlegungen zu Antwortzeiten beziehen sich nun aber auf die reale Durchführung von Lösungsprozessen in interaktiven Problemlösungen. Bei der Bildung eines Modells für eine interaktive Problemlösung ist die Zeit[2] in erster Linie[3] zur Bestimmung der Reihenfolge von Lösungsschritten - vorher, gleichzeitig, nachher - wesentlich; hier sind nur die Abhängigkeiten vorher und nachher zu berücksichtigen, da Gleichzeitigkeit aus der Betrachtung ausgeschlossen worden ist vgl. 2.2.(vi). Es erscheint daher

---

[1] Vgl. dazu die experimentellen Studien zu "on-line versus off-line problem solving" in (SACKMAN,1970), die tendenziell die Aussage erlauben, daß "the batch mode favors more economicall machine performance whereas time-sharing favors more effective and more satisfactory human performance with a higher-quality end product" (p.119); allerdings ist die Abhängigkeit solcher Aussagen von den gelösten Problemen, Benutzerkreis, u.ä. zu beachten.

[2] Hier genauer: ein Modell für Zeit, bzw. Zeitpunkte, etwa in Gestalt der reellen, bzw. natürlichen Zahlen.

[3] Programmierte Zeitsteuerung, wie sie z.B. in BASIC (vgl. UNIVAC,1972) die Anweisung WAIT ermöglicht, kann hier, aufgrund von Voraussetzung 2.2.(vi) vernachlässigt werden.

gerechtfertigt, im Modell "interaktive Prozedur" die Zeit nur implizit durch die im Ableitungs- (Def.2.3.1.2.) und Lösungsprozeßbegriff (Def.2.3.1.3.) enthaltenen Vorher-Nachher-Beziehungen[1] einzubeziehen. Dementsprechend sind - von der Zeit her gesehen - mit einer interaktiven Prozedur interaktive Problemlösungen beschreibbar mit ausschließlich sequentiellen Lösungsprozessen, die sowohl in "on-" wie "off-line-Kommunikation" - ggf. auch durcheinander - ablaufen können.

Lösungsprozesse in interaktiven Prozeduren sind dann synchron zur Zeit[2] und können nicht parallel verlaufen, so daß hier - vom Modell her gesehen - keine Synchronisationsprobleme von Lösungsprozessen auftreten[3].

---

[1] Ist eine Konfiguration x' aus einer Konfiguration x ableitbar, so ist x' zeitlich als nach x und x als vor x' entstanden anzusehen.

[2] Vgl. die Indexmengen von Interaktionen in Def.2.3.1.3.

[3] Synchronisationsprobleme treten bei expliziter Berücksichtigung von Gleichzeitigkeit von Lösungsprozessen auf, vgl. dazu die Diskussion von Modellen für interaktive Problemlösungen in 2.3.1.

3. Struktur und Eigenschaften interaktiver Prozeduren

Es werden nun Struktur und Eigenschaften interaktiver Prozeduren mit dem Ziel untersucht, damit Aussagen über Struktur und Eigenschaften interaktiver Problemlösungen zu gewinnen. Dabei ist zu berücksichtigen, daß die vorgenommene Modellbildung die Realität interaktiver Problemlösungen in Bezug auf die Beschreibungsmerkmale aus 2.2. wiedergibt.

3.1. Dialoge

Als eine erste Anwendung der in 2.3. durchgeführten Modellbildung wird die Möglichkeit exakter Begriffsbildung, im folgenden für Dialoge, betrachtet.

3.1.1. Formale Definition von Dialogen

In einem Lösungsprozeß in einer interaktiven Prozedur ist nach den bisherigen Betrachtungen ein Kommunikationsprozeß zwischen Benutzer und Rechner nicht explizit festgelegt. Dies liegt daran, daß in Lösungsprozessen in einer interaktiven Prozedur T zwar die Aktivitäten eines Benutzers als "eingegebene Zeichenkette" nach einem Unterbrechungszustand explizit vorhanden sind, jedoch in einem Unterbrechungszustand noch festzulegen ist, welche Information aus dem bis

dahin geführten Lösungsprozeß in T als eine dem Benutzer erkennbare Aktivität des Rechners angesehen werden soll.

Seien dazu $T = (Q, Q_U, A, d, q_0)$ eine interaktive Prozedur und $t = \{(q^{(i)}, w^{(i)}, a(w^{(i)}))\}_{i \in I}$, $I=\{0,1,..\}$ Indexmenge von t, ein Lösungsprozeß in T. Zu t sei dann

$W_t := \{(w^{(0)}, \ldots, w_{j_i}, w_{j_{i+1}}, \ldots) : w_{j_i}$ Teilwort aus $w^{(i)}$, $w_{j_{i+1}} \in A^+$ mit "$w_{j_{i+1}}$ wird in $q^{(i)}$ ausgewählt"[1], $i \in I$ und $q^{(i)} \in Q_U - \{q_0\}$ oder, $i=n$ und $I=\{0,1,..,j_n\}\}$.

W_t ist nichtleer für jeden Lösungsprozeß t in T und bezeichnet die Menge zur Durchführung von t möglicher Kommunikationsprozesse an Unterbrechungszuständen zwischen Benutzer und Rechner. Sei nun

$$W_T := \bigcup_{\substack{t:t \text{ Lösungs-} \\ \text{prozeß in } T}} W_t \ ;$$

dann heiße eine Abbildung

$diag_T : \{t: t \text{ Lösungsprozeß in } T\} \longrightarrow W_T$,

die $diag_T(t) \in W_t$ für alle Lösungsprozesse t in T erfüllt, eine Dialogmarkierung in T.

Definition 3.1.1.1.: Seien $T = (Q, Q_U, A, d, q_0)$ eine interaktive Prozedur und $diag_T$ eine Dialogmarkierung in T. Dann heiße $diag_T(t)$ für einen Lösungsprozeß t in T der in t geführte Dialog[2].

---

[1] D.h.: es ist $d(q^{(i)}, a(w^{(i)}) = \{(q_{i_1}, b_{i_1}, A_{i_1}), .., (q_{i_r}, b_{i_r}, A_{i_r})\}$ $w \in A_{i_j} : q_{i_j} = q^{(i+1)}$ und w wird bei Ableitung von $(q^{(i+1)}, w^{(i+1)}, a(w^{(i+1)}))$ aus $(q^{(i)}, w^{(i)}, a(w^{(i)}))$ in $w^{(i)}$ an der Stelle $a(w^{(i)})$ entsprechend Def.2.3.1.2. eingesetzt.

[2] Zum Begriff Dialog vgl. 2.2.(i); vgl. auch (KUPKA-WILSING, 1975), p.119,120, u.a.

Ein Dialog besteht also aus den zu Beginn eines Lösungsprozesses vorhandenen Informationen, einer abwechselnden Folge von Ausschnitten der an Unterbrechungszuständen vorhandenen Informationen sowie den dort erfolgenden Aktivitäten eines Benutzers und Teilen der Endinformation, sofern der Lösungsprozeß endlich ist. Die Konstruktion der Abbildung $diag_T$ spiegelt dabei wider, daß bei der Konzeption einer interaktiven Problemlösung an jeder Unterbrechung festzulegen ist, welche der bis dahin im Rechner vorhandenen Informationen zur Unterrichtung eines Benutzers vom Rechner bereitgestellt werden sollen. Weiterhin ist in $diag_T$ das "Design von Dialogen"[1] innerhalb interaktiver Lösungsprozesse enthalten, von dem wesentlich deren Effizienz (Anzahl der Lösungsschritte, "Qualität" einer Lösung, u.ä.) abhängen kann; dabei muß $diag_T$ auch hardwaremäßige Gegebenheiten berücksichtigen, etwa wenn T eine Problemlösung beschreibt, die gewisse Anforderungen an die Hardware stellt ("APL-Design"[2], u.ä.), oder T generell auf einem (stets) gewisse Beschränkungen auferlegenden speziellen realen Rechner implementiert werden soll. Beschreibt T interaktives Problemlösen in einer Dialogsprache[3], so muß $diag_T$ deren Sprach-

---

[1] Für einen Überblick über Faktoren, die das "Design von Dialogen" bestimmen, siehe (MARTIN,1973), p.10, Figur 2.1.

[2] Man vergleiche etwa die Tastatur eines "APL-Terminals" in (GILMAN-ALLEN,1973), p.20, mit ihren speziellen Symbolen mit deren "Umsetzungen" in (UNIVAC,1973) auf die allgemeine alphanumerische Tastatur etwa des Datensichtgerätes UNISCOPE 200 (UNIVAC).

[3] Für eine Beschreibung von Möglichkeiten zum Problemlösen, die einige Dialogsprachen bieten, siehe etwa (KUPKA-WILSING,1975), Teil 3., (SCHUR,1973).

elemente im wesentlichen beinhalten[1].

Es ist klar, daß innerhalb einer interaktiven Prozedur, die keine Unterbrechungszustände außer höchstens q_0 hat, nur triviale Dialoge der Form $w^{(0)}$, $w^{(0)} \in A^+$ Startwort, oder $(w^{(0)},w')$, w' Teil des Resultats einer Interaktion, führbar sind. Dies entspricht im Rahmen des Modells der Situation, daß Programme von einem Rechner mit gegebenen Daten $w^{(0)}$ selbsttätig durchgeführt werden, nachdem ein derartiger Lösungsprozeß von einem Benutzer höchstens gestartet wurde.

Ein Beispiel, aufbauend auf der interaktiven Prozedur aus 2.3.3. zur Transportmitteldisposition soll nun die Darstellung von Dialogen verdeutlichen.

3.1.2. Dialoge innerhalb der interaktiven Transportmitteldisposition

Es wird wieder der Problemlösungsausschnitt aus Teil 2.3.2. der interaktiven Transportmitteldisposition betrachtet. Eine in der dort angegebenen interaktiven Prozedur T durchführbare Interaktion ist:

t_0 := $(q_0, _\cdots_, _)$[2]

$(q_1, _\cdots$WEITERE_TRANSPORTMITTELVERGABE_GEWUENSCHT?$_\cdots_, _)$

$(q_2, _\cdots_^{\alpha}$[3]$H_\cdots_, H)$

---

[1] T muß dann offenbar die "Verarbeitung" dieser Sprachelemente beinhalten; für eine Diskussion derartiger Verarbeitungsformen (interpretativ, compilierend, u.ä.) siehe (SCHMID, 1974).

[2] Ein Index der aktuellen Leerstelle "_" werde vernachlässigt.

[3] Zur Definition von α vgl. 2.3.3.

(q_0, _..._ LETZTE_ANTWORT_FEHLERHAFT,_DAHER_NICHT_
BERUECKSICHTIGT._..._, _)

(q_1, _..._$\alpha\beta$[1])$_\alpha$_..._, _)

(q_2, _..._$\alpha\beta\alpha$J_..._, J)

(q_{21}, _..._$\alpha\beta\alpha$J_..._, J).

Eine weitere Fortsetzung dieses Lösungsprozesses ist nicht möglich, da in T wegen der lokalen Betrachtung in 2.3.3. keine aus der letzten Konfiguration von t_0 ableitbaren Konfigurationen existieren. t_0 wird nun die Menge

W_{t_0} := {((_)^m, w_1, H, w_2, J) , w_1 Teilwort aus _.._α_.._ ,
w_2 Teilwort aus _..._$\alpha\beta\alpha$_..._ } ,

m hinreichend große Zahl, (vgl.2.3.3.), von Wortfolgen zugeordnet.

Sei nun

$$diag_T(t) := \begin{cases} ((_)^m, \alpha, H, \beta\alpha, J), \text{ falls } t = t_0, \\ w_t \in W_t, \text{ falls } t \neq t_0 \text{ [2)]}, \end{cases}$$

so kann $diag_T(t_0)$ mit dem gekennzeichneten Dialogauschnitt im Dialogprotokoll von Figur 3.1.2.1. identifiziert werden; das Dialogprotokoll aus Figur 3.1.2.1. ist die Auflistung eines Dialoges zwischen Benutzer und Rechner in einem Lösungsprozeß in der implementierten interaktiven Transportmitteldisposition[3)] aus 2.1..

[1)] Zur Definition von β vgl. 2.3.3.

[2)] Explizit interessiere hier nur $diag_T(t_0)$.

[3)] Vgl. Anhang 1.

```
WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN?  H
LETZTE ANTWORT FEHLERHAFT,DAHER NICHT BERUECKSICHTIGT.
WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN?  J
MOEGLICHE TRANSPORTMITTEL, (RED.)KAPAZITAETEN, KOSTENSAETZE:
  T  1 T  2 T  3 T  4
  450 1200  800  350
   12   14   13   16
KUNDEN, (RED.)BESTELLMENGEN, ENTFERNUNGEN:
  K  1 K  2 K  3 K  4 K  5
  300  150  640  980  670
  240  870  400  630  520
VERGEBEN SIE (WEITERE) TRANSPORTMITTEL DURCH ANGABEN UNTER:
TRANSP.MITTELNR.    KUNDENNR.    ZU TRANSP. MENGE
    2                   4                80
WEITERE TRANSPORTMITTELVERGABE GEWUENSCHT?  N
```

Figur 3.1.2.1.[1)]

Aufgrund der lokalen Betrachtung wurde hier zusätzlich ein zu Beginn von t_0 "leerer Bildschirm" und ein Ende des Lösungsprozesses nach der Verzweigung zu den möglichen Lösungsprozeßfortsetzungen TRANSPORTMITTELVORGABE und TRANSPORTVERGABE (in der implementierten Lösung) angenommen; diese Annahmen sind bei umfassenderer Betrachtung, etwa aller am Dialog in Figur 3.1.2.1. beteiligten Problemlösungskomponenten der Transportmitteldisposition, zu modifizieren.

---

1) Die Zahlen sind Werte für Variable aus der Problemlösung der Transportmitteldisposition, die im Lösungsprozeß mit dem aufgelisteten Dialog zugrunde lagen; sie sind hier ohne Bedeutung. "T i", i=1,..,4, sind die angebotenen Transportmittel, "K j", j=1,..,5, die in τ zu beliefernden Kunden.

3.2. Zur Struktur interaktiver Prozeduren

Auf 2.3.2. aufbauend wird nun die Struktur interaktiver Prozeduren weiter betrachtet. Es werden Grundkomponenten interaktiver Prozeduren herausgearbeitet; diese sind Basis für anschließende strukturbezogene Überlegungen zur Implementierung und Dokumentation interaktiver Problemlösungen.

3.2.1. Grundkomponenten interaktiver Prozeduren

In Satz 2.3.2.1. wurde gezeigt, daß man nicht von Unterbrechungen unterbrochene Teile einer interaktiven Prozedur, die in gewissem Sinne[1] zusammenhängen, als Turingprogramme auffassen kann. Darauf aufbauend werden nun Grundkomponenten interaktiver Prozeduren bestimmt; mit ihrer Hilfe lassen sich u.a. ein "user interface", dies sei eine problemabhängige Beschreibung der führbaren Dialoge und der dabei möglichen Resultate, gewinnen.

Dazu sind einige Vorbetrachtungen erforderlich. Seien $T = (Q, Q_U, A, d, q_O)$ eine beliebige, in diesem Teil feste interaktive Prozedur[2] und

---

[1] Dieser wird in Satz 2.3.2.1. durch die Menge $W(q,q')$ gegeben.

[2] Auf eine Indizierung der Zustandsmengen u.ä. von T wird dementsprechend im folgenden verzichtet.

$Q_F := \{q \in Q : \exists$ Interaktion $t = \{(q^{(i)}, w^{(i)}, a(w^{(i)}))\}_{i=0,..,n(t)}$
und $q^{(n(t))} = q\}$.

Seien nun $Q' := \{q_0\} \cup \bigcup_{q \in Q_U} N(q)$ [1]

und $Q'' := Q_U \cup Q_F$.

Dann gilt:

Satz 3.2.1.1.: Sei $T = (Q, Q_U, A, d, q_0)$ eine interaktive Prozedur; dann ordnet das Verfahren

(i) bilde Q' und Q'',

(ii) bilde zu jedem Zustandspaar $(q',q'') \in Q' \times Q''$ die Menge $W(q',q'')$ [2],

(iii) bilde zu jeder nichtleeren Menge $W(q',q'') = \{q_{i_1}=q', \ldots, q_{i_r(q',q'')}=q''\}$ das Turing- [3] programm

$$P_{q',q''} := \begin{cases} \{B_{W(q',q'')}(q_{i_1}), .., B_{W(q',q'')}(q_{i_{r-1}}), \emptyset\}, & q'' \in Q_U, \\ \{B_{W(q',q'')}(q_{i_1}), .., B_{W(q',q'')}(q_{i_r})\}, & q'' \notin Q_U, \end{cases}$$

T eindeutig eine endliche Familie von Turingprogrammen $\{P_{q',q''}, (q',q'') \in Q' \times Q'' : W(q',q'') \neq \emptyset\}$ zu.

Der Beweis von Satz 3.2.1.1. ist einfach.

Es ist möglich, daß die bestimmte Familie von Turingprogrammen $\{P_{q',q''}\}$ leer ist - etwa wenn Q_U leer ist und kein

---

[1] N(q) bezeichne die Menge aller direkten Nachfolgezustände in G(T) des Zustandes q; zum Begriff des direkten Nachfolgers siehe (NOLTEMEIER,1976), p.26.

[2] Zur Bildung der Mengen W(q',q'') vgl. 2.3.2.

[3] Das Vorliegen eines Turingprogrammes besagt Satz 2.3.2.1.; zur Bildung der Befehlsmengen $B_{W(q',q'')}(q_{i_k})$ vgl. 2.3.2.

Lösungsprozesse endlicher Länge in T führbar sind; im Regelfall wird eine interaktive Problemlösung einige wenige Stellen enthalten, die ein "reguläres Beenden" von Lösungsprozessen - das heiße in einem Zustand aus Q_F - ermöglichen[1]. Das Verfahren aus Satz 3.2.1.1. schließt nicht notwendig alle Zustände derart ein, daß sie einen nichtleeren Befehl in einem der Turingprogramme $P_{q',q''}$ bestimmen; solche Zustände sind jedoch in keiner, innerhalb T durchführbaren Interaktion erreichbar und bilden Problemlösungsteile, die von keinem in T führbaren Lösungsprozeß erreicht werden können, oder jeden sie erreichenden Lösungsprczeß zu einem beliebiger Länge werden lassen(nicht endliche Indexmenge in Def.2.3.1.3.). Es ist klar, daß das Vorkommen solcher Zustände[2] in einer interaktiven Prozedur erfordert, festzustellen, ob sie überflüssig sind oder eine fehlerhafte Konzeption des Problemlösungsmodells T anzeigen - bezüglich des zu lösenden Problems.

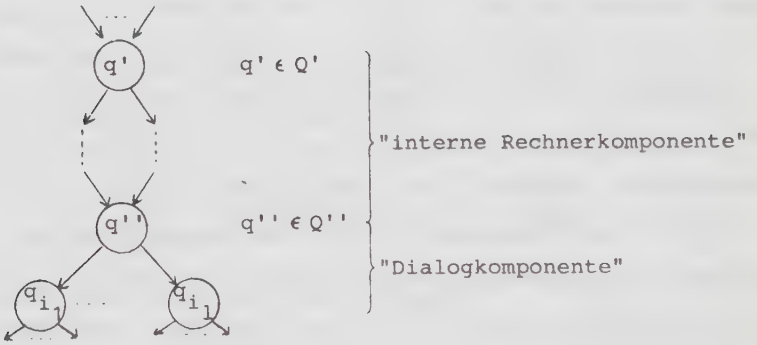
Setzt man nun voraus, daß T keine der angegebenen Problemzustände enthält, d.h., jeder Zustand aus T eine nichtleere Befehlsmenge in einem Turingprogramm aus $\{P_{q',q''}\}$ bestimmt, so hat Satz 3.2.1.1. einige interessante Konsequenzen:

---

[1] Man beachte, daß dabei implizit vorausgesetzt ist, daß ein Benutzer in einem Unterbrechungszustand stets eine Aktivität ergreift.

[2] Bei vorgegebener interaktiven Prozedur T können solche Zustände mit aus der Programmoptimierung bekannten Methoden festgestellt werden, soweit sie aus der Struktur des Transitionsgraphen G(T) bestimmt werden können; für in der Programmoptimierung behandelte Fragestellungen und Literatur dazu siehe (ROTH,1975c).

1. T besteht aus Grundkomponenten der folgenden Art (eine Bewertung der Pfeile ist zur besseren Übersichtlichkeit weggelassen):



Figur 3.2.1.1.

Als "interne Rechnerkomponente" werde das Turingprogramm $P_{q',q''}$ bezeichnet. Ist $q'' \in Q_U$, so besteht die "Dialogkomponente" aus der Auswahl eines Wortes aus dem im Zustand q'' einem Benutzer zur Verfügung stehenden Spielraum an Worten - mit ihr ist dann eindeutig, vgl. Def. 2.3.1.1.(iii), die Wahl eines Folgezustandes aus $q_{i_1},\ldots,q_{i_l}$ verbunden - und dessen Einfügen in der in Def. 2.3.1.2. beschriebenen Weise an die aktuelle Stelle der im Zustand q'' in T bis dahin erarbeiteten Information (Wort). Ist $q'' = q_o$ oder $q'' \in Q_F - Q_U$, so reduziert sich die Dialogkomponente auf deterministische Zustandsübergänge ohne Benutzereinwirkung. Hat q'' keinen direkten Folgezustand, so entfällt die Dialogkomponente.

2. $|\{P_{q',q''} , (q',q'') \in Q' \times Q'' : W(q',q'') \neq \emptyset\}|$ ist - abgesehen von der Implementierung der Dialogkomponenten - die Maximalzahl zu implementierender unterschiedlicher interner Rechnerkomponenten; die Semantik jeder dieser internen Rechnerkomponenten wird dabei durch das zugehörige Turingprogramm in operationeller[1] Form gegeben.

Bei einer Implementierung von Grundkomponenten von T der in Figur 3.2.1.1. gegebenen Form auf einem realen Rechner in einer nichtdialogspezifischen Programmiersprache - etwa in ALGOL wie die interaktive Transportmitteldisposition aus 2.1.[2] - wird eine Dialogkomponente durch Ein-/Ausgabeelemente und eine interne Rechnerkomponente durch nicht Ein-/Ausgabespezifische Elemente zur Manipulation von Daten der Sprache gegeben. Bei einer Implementierung in einer Dialogsprache können, in der Terminologie von (KUPKA-OBERQUELLE-WILSING, 1974), eine interne Rechnerkomponente als ein mit Elementen des Kernes und eine Dialogkomponente als ein mit Elementen des Rahmens einer Dialogsprache[3] zu realisierender Lösungsteil aufgefaßt werden.

---

[1] Zum Begriff "operationelle Semantik" und dem Zusammenhang dieser Art einer Semantikdefinition zu weiteren siehe (WEGNER,1972),p.128ff.

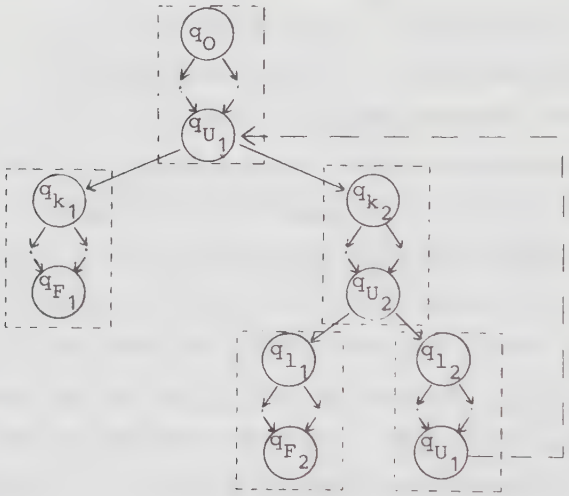
[2] Die Programmkomponenten der interaktiven Transportmitteldisposition sind im ALGOL-Dialekt NUALGOL, vgl. (UNIVAC, 1971), geschrieben.

[3] Vgl.(KUPKA-OBERQUELLE-WILSING,1974),p.7ff.

Gilt für alle Befehle $B_{W(q',q'')}(q)$ aller Turingprogramme $P_{q',q'''}$, daß

$$B_{W(q',q'')}(q) = B_Q(q) \ ,$$

$q \in W(q',q'')$, so hat T offenbar eine besonders durchsichtige Struktur, wie etwa im folgenden Beispiel:



$Q_F = \{q_{F_1}, q_{F_2}\}$, $Q_U = \{q_{U_1}, q_{U_2}\}$.

Figur 3.2.1.2.

Eine Implementierung von T in Form von Makros[1], die jeweils eine interne Rechnerkomponente darstellen, ist dann naheliegend. Ein Beispiel für eine derartige Implementierung ist das bereits in 1.2.2.2. angesprochene Lösen von Zuordnungsproblemen in (LOSBICHLER-MÜHLBACHER,1974).[2]

---

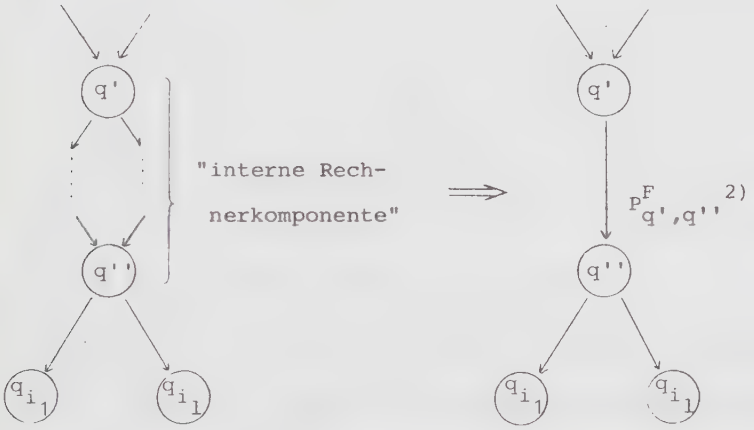
[1] Zum Begriff Makro siehe (CAMPBELL,1973), insbesondere Teil 6.

[2] Vgl. insbesondere die Beschreibung eines Interpreters für die verwendeten Makros (LOSBICHLER-MÜHLBACHER,1974), p.458ff.; für generelle Überlegungen zum Einsatz compilierender und interpretierender Systeme in Implementierungen interaktiver Problemlösungen siehe (SCHMID,1974), insbesondere Teil 3.

3. Jedes der Turingprogramme $P_{q',q''}$, in die T abgesehen von Dialogkomponenten zerfällt, stellt eine partielle Abbildung

$$P^F_{q',q''} : DEF(P_{q'q''}) \rightarrow A^+ \times \mathbb{N}$$

dar; dabei ist $DEF(P^F_{q',q''})$ die Menge der Informationen $(w,n) \in A^+ \times \mathbb{N}$ [1], für die ein Lösungsprozeß $t=\{(q^{(i)},w^{(i)}, a(w^{(i)}))\}_{i \in I}$, $I=\{0,1,\ldots\}$ existiert mit $q^{(i_0)}=q'$, $w^{(i_0)}= w$ und $a(w^{(i_0)}) = w_n$ für ein $i_0 \in I$. Damit läßt sich eine "interne Rechnerkomponente" (vgl. Figur 3.2.1.1.) in der folgenden Weise zusammenfassen:



Figur 3.2.1.3.

---

[1] Man beachte, daß zwischen dem Index des aktuellen Zeichens und dem Zeichen selbst im allgemeinen nicht unterschieden wird.

[2] Man beachte, daß die Befehle von $P_{q',q''}$ bezüglich W(q',q'') -vgl. Satz 3.2.1.1. - definiert sind.

4. Für eine interaktive Prozedur T kann man nun ohne Beschränkung der Allgemeinheit voraussetzen, daß sie höchstens[1] einen Zustand q_F mit $B_Q(q_F) = \emptyset$ besitzt, in dem alle Interaktionen in T enden. Dann läßt sich T in naheliegender Weise zu einer funktionsorientierten Darstellung T^F zusammenfassen[2] - aufbauend auf der oben in 3. abgeleiteten funktionalen Darstellung von Grundkomponenten einer interaktiven Prozedur. Die funktionsorientierte Darstellung T^F des Modells T einer interaktiven Problemlösung hat folgende Bedeutungen:

(i) Sie erlaubt eine benutzerorientierte Darstellung der von T modellhaft beschriebenen interaktiven Problemlösung von hohem Dokumentationswert; insbesondere können leicht Veränderungen bis hin zu Erweiterungen von T mit ihren Konsequenzen, die sie für die gesamte Problemlösung T haben, übersehen werden. Zudem ist ein Überprüfen der "Korrektheit" von T unter vorgegebenen Bedingungen, wenn man von "korrekten" internen Rechnerkomponenten ausgeht, in T^F leicht möglich, was in 3.4. präzisiert wird. Für den "Designer" einer interaktiven Problemlösung lassen sich anhand von T^F leicht Fehler in der Gesamtkonzeption der Problemlösung feststellen.

---

[1] In einer interaktiven Prozedur müssen nicht notwendig Lösungsprozesse endlicher Länge existieren.

[2] Vgl. hinten im Beweis von Satz 3.3.2.1. die Betrachtung der funktionalen Darstellung von Grundkomponenten von T.

(ii) Betrachtet man T^F, eingeschränkt auf die in T führbaren Dialoge (gegeben durch $diag_T$, vgl. 3.1.1.), so erhält man eine dialogorientierte Darstellung von T in Art eines "terminal state transition diagram " aus (PARNAS,1968)[1]; die Definition einer interaktiven Prozedur T sowie in ihr führbarer Dialoge ermöglichen den "reponses to vary within a single (terminal) state"[2], was in (PARNAS,1968) als wesentliche Eigenschaft der dort formal nicht näher spezifizierten "terminal states" gefordert wird. Eine dialogorientierte Darstellung von T erlaubt nun einem Benutzer in besonderem Maße, sich mit den in T führbaren Dialogen vertraut zu machen. Ist der Benutzer derjenige, der eine interaktive Problemlösung erstellt, so erlaubt ihm die dialogorientierte Darstellung von T eine Überprüfung, in wieweit er die Arbeitsmöglichkeiten realisiert hat, die spätere Benutzer der interaktiven Problemlösung haben sollen[3].

Generell kann vom Betrachten der Struktur von T^F eher eine Rückkopplung auf die Konzeption von T ausgehen als von T selbst, da T^F im Vergleich zu T eine leichter überschaubare Grobstruktur aufweist.

---

[1] Vgl. (PARNAS,1968), p. 380.

[2] Vgl. (PARNAS,1968), p. 380.

[3] Vgl. die "Applications of Terminal State Transition Diagrams" in (PARNAS,1968), p. 382ff.

Einige der Überlegungen dieses Abschnittes sollen nun wieder innerhalb der interaktiven Transportmitteldispositions-Problemlösung aus 2.1. verdeutlicht werden.

3.2.2. Grundkomponenten eines Ausschnitts der interaktiven Transportmitteldisposition

Es werde zunächst wieder der schon in 2.3.3. behandelte Ausschnitt T der interaktiven Transportmitteldisposition betrachtet. Für diesen Ausschnitt sind $Q' = \{q_o, q_2\}$ und $Q'' = \{q_1, q_{21}, q_{22}\}$ mit $Q_U = \{q_1\}$ und $Q_F = \{q_{21}, q_{22}\}$[1]. Es ergeben sich damit die folgenden Mengen $W(q',q'')$, $(q',q'') \in Q' \times Q''$:

$W(q_o,q_1) = \{q_o,q_1\}$, $\quad W(q_o,q_{21}) = \emptyset$,

$W(q_o,q_{22}) = \emptyset$, $\quad W(q_2,q_1) = \{q_2,q_o,q_1\}$,

$W(q_2,q_{21}) = \{q_2,q_{21}\}$, $\quad W(q_2,q_{22}) = \{q_2,q_{22}\}$,

die nach Satz 3.2.1.1. T eindeutig eine Familie

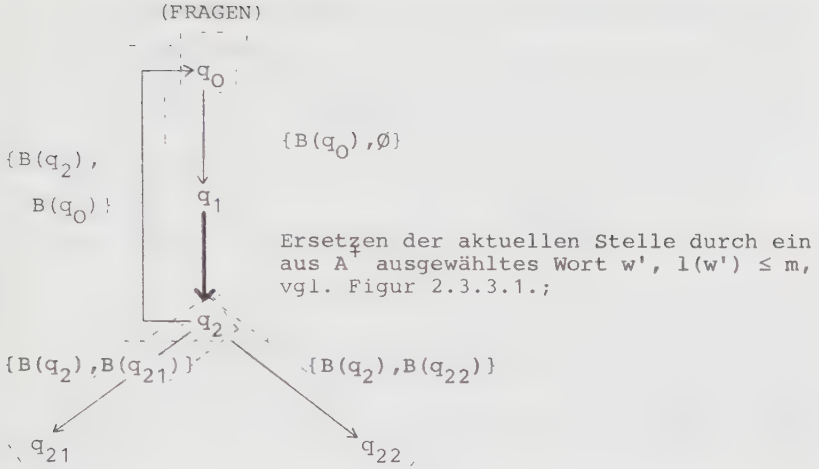
$\{ \{B(q_o),\emptyset\}, \{B(q_2),B(q_o)\}, \{B(q_2),B(q_{21})\}, \{B(q_2),B(q_{22})\} \}$[2]

von Turingprogrammen als Grundkomponenten zuordnen. Damit zerfällt T in die folgenden Komponenten - die internen Rechnerkomponenten sind in der folgenden Figur gestrichelt umfaßt, die Dialogkomponenten verstärkt gezeichnet:

---

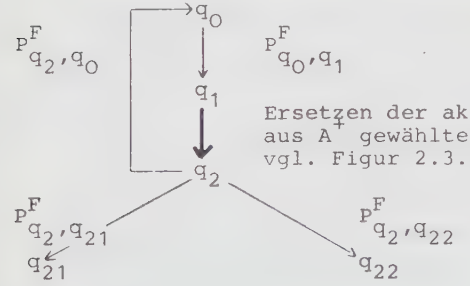
[1] Da hier nur ein Ausschnitt aus der interaktiven Transportmitteldisposition betrachtet wird, wird hier nicht die Voraussetzung aus 3.2.1.4. erfüllt.

[2] Da eindeutig ist, zu welchen Mengen $W(q',q'')$ die einzelnen Turingprogramme gebildet sind, werden die einzelnen Befehle nicht mit den entsprechenden Zustandsmengen indiziert; für ihre Definition vgl. 2.3.2.



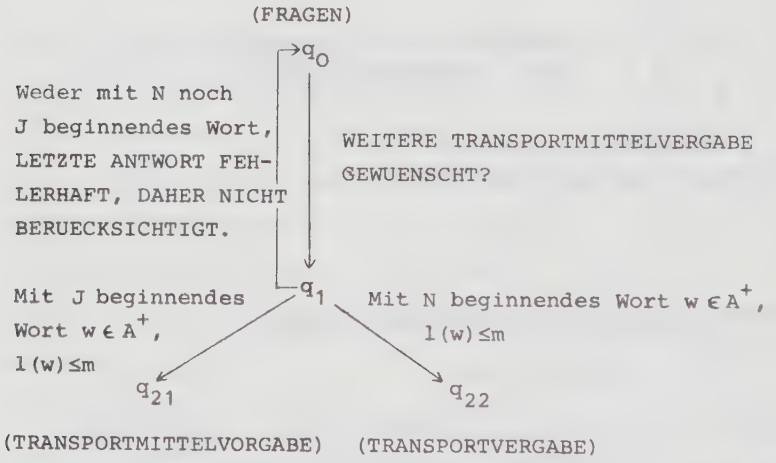
Figur 3.2.2.1.

Der Übergang zur funktionsorientierten Darstellung T^F von T ergibt dann die folgende Darstellung des Übergangsverhaltens von T (vgl. 3.2.1.3., 3.2.1.4.):



Figur 3.2.2.2.

Aus Figur 3.2.2.2. kann man dann leicht eine dialogorientierte Darstellung der in T verfügbaren Arbeitsmöglichkeiten gewinnen (vgl. 3.2.1.4.). Die im folgenden Diagramm verwendete Dialogmarkierung erlaubt das Führen der Dialoge aus 3.1.2.



Figur 3.2.2.3.

Vergleicht man die Darstellungskonzepte aus Figur 3.2.2.2. und Figur 3.2.2.3. mit dem globalen Lösungsablaufplan der Transportmitteldisposition aus Figur 2.1.1.1., so kann dieser - in verbaler Form - als eine funktionsorientierte Darstellung angesehen werden, in der die Semantik der einzelnen Problemlösungskomponenten durch Turingprogramme für die internen Rechnerkomponenten und durch die Auswahl- und Einsetzungsvorschriften für die Dialogkomponenten gegeben wird. Durch Weglassen der internen Rechnerkomponenten und ihre indirekte Einbeziehung in eine dialogorientierte Beschreibung erhält man die Transportmitteldisposition in Gestalt von Figur 3.2.2.3.

3.3. Interaktive Prozeduren und nichtdeterministische Programme

Die funktionsorientierte Darstellung T^F einer interaktiven Prozedur aus 3.2.1. legt nun nahe, Zusammenhänge zwischen interaktiven Prozeduren und den auf Floyd,(FLOYD,1967a), zurückgehenden nichtdeterministischen Programmen zu bestimmen.

3.3.1. Auswählende Programme

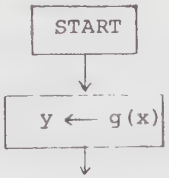
In Anlehnung an (MANNA,1970),(NILSSON,1971), werde definiert:

Definition 3.3.1.1.: Ein auswählendes Programm AP bestehe aus

(i) einer Eingabevariablen x, Programmvariablen y und Ausgabevariablen z mit nichtleeren Definitionsbereichen D_x, D_y und D_z;

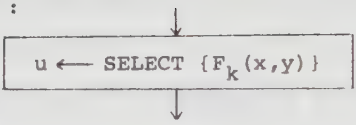
(ii) einem Flußdiagramm, das sich aus folgenden Anweisungskomponenten zusammensetzt:

1. START :



g ist eine partielle Abbildung von D_x in D_y; diese Komponente komme genau einmal vor;

2. SELECT :

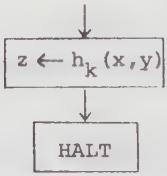


F_k ist eine partielle Abbildung von $D_x \times D_y$ in $P(D_u)-\emptyset$, mit $k \in I(AP)$, $I(AP)$ endliche Indexmenge[1], $u \in \{x,y\}$;

3. $\bigvee$-BRANCH :

$p_{i_1},\ldots,p_{i_n}$ sind totale Prädikate[2] über $D_x \times D_y$;

4. HALT :



h_k ist eine partielle Abbildung von $D_x \times D_y$ in D_z, $k \in I(AP)$.

---

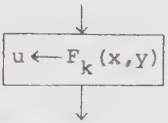
[1] $I(AP)$ enthält die Indizes aller SELECT-Anweisungen sowie Zuordnungsanweisungen vor einer HALT-Anweisung von AP.

[2] Zum Begriff Prädikat siehe (HERMES,1972),p.44ff.; total bedeute, daß die Prädikate auf ganz $D_x \times D_y$ erklärt sein sollen.

Die hier definierten auswählenden Programme[1] können als nichtdeterministische Programme im Sinne von (MANNA,1970), (NILSSON,1971) aufgefaßt werden, wenn man von der dortigen Forderung nach totalen Abbildungen in 1.,2.und 4.absieht und unterstellt, daß das Erreichen von HALT stets als ein "Erfolg"[2] anzusehen ist.

Ein Beispiel eines auswählenden Programmes ist etwa das Ordnen eines "Zahlenpuzzles" in (NILSSON,1971)[3].

Offenbar ist für den Fall, daß F_k in einer SELECT-Anweisung eine Abbildung in D_u ist, die übliche Zuordnungsanweisung



eingeschlossen[4].

---

[1] Dieser Name wurde gewählt, da auswählende Programme im Vergleich zu nichtdeterministischen Programmen nur Anweisungen zum Auswählen (von Elementen oder Lösungsprozeßfortsetzungen) enthalten, nicht jedoch Anweisungen zum parallelen Fortsetzen von Lösungsprozessen, wie etwa " ∀-SELECT" und "∧-BRANCH" in (MANNA,1970),p.5.

[2] Vgl. "Success-Halt" in (MANNA,1970),p.5.

[3] Vgl. (NILSSON,1971),p.26; für weitere Beispiele siehe (FLOYD,1967a),p.637, (MANNA,1970),p.10.

[4] Vgl. (MANNA,1970),p.5.

Für den Fall, daß in einer $\bigvee$-BRANCH-Anweisung n gleich 2 ist und

$$p_1(x,y) = \begin{cases} T^{1)} & \text{, falls } p(x,y) \\ F, \text{ falls } \neg p(x,y)^{2)} \end{cases} \qquad p_2(x,y) = \begin{cases} T, \text{ falls } \neg p(x,y) \\ F, \text{ falls } p(x,y) \end{cases}$$

für ein totales Prädikat $p(x,y)$ über $D_x \times D_y$ gilt, ist die übliche Verzweigungsanweisung für Programme eingeschlossen. Auswählende Programme verallgemeinern also durch Flußdiagramme gegebene Programme "üblicher Art".

Seien nun AP ein auswählendes Programm und $\xi \in D_x$; eine Ausführung von AP mit ξ geschieht dann in der durch das Flußdiagramm gegebenen Reihenfolge dadurch[3)], daß

- von der START-Anweisung x einen Wert $\xi \in D_x$ und y den Wert $g(\xi)$ zugeordnet bekommen (falls $g(\xi)$ definiert ist[4)]);
- von einer SELECT-Anweisung u ein aus $F_k(\xi,\eta)$ ausgewähltes Element zugeordnet wird (falls $F_k(\xi,\eta) \neq \emptyset$ ist), wobei ξ der Wert von x und η der Wert von y vor dieser Anweisung seien, $u \in \{x,y\}$;
- an einer $\bigvee$-BRANCH-Anweisung zu einer der direkten Folgeanweisungen verzweigt wird, für die das entsprechende Prädikat $p_i(\xi,\eta)$ gilt, wobei ξ der Wert von x und η der von y vor dieser Anweisung seien;

---

1) Die Wahrheitswerte wahr und falsch werden mit T bzw. F bezeichnet.

2) Der Operator "$\neg$" bezeichne die logische Verneinung.

3) Vgl. (MANNA,1970),p.6ff.

4) Ist $g(\xi)$ bzw. $F_k(\xi,\eta)$ bzw. $h_k(\xi,\eta)$ nicht definiert, so breche die Ausführung ohne Resultat ab.

- an einer HALT-Anweisung z der Wert $h_k(\xi,\eta)$ zugeordnet wird, wobei ξ der Wert von x und η der von y vor dieser Anweisung seien (falls $h_k(\xi,\eta)$ definiert ist).

Die Folge der sich bei einer Ausführung eines auswählenden Programmes mit einem Startwert $\xi \in D_x$ ergebenden Werte der Variablen x, y und z an START-, SELECT- und HALT-Anweisungen soll als ein Ausführungsweg $p(\xi)$ von AP bezeichnet werden; ein Ausführungsweg habe das Resultat η, falls bei seiner Ausführung eine HALT-Anweisung erreicht wird, und z dabei der Wert η zugewiesen wurde.
Damit definiert AP eine partielle Abbildung

$$AP^F : D_x \rightarrow P(D_z)$$

definiert durch $AP^F(\xi) := \{\eta : \eta \in D_z$ und η Resultat eines Ausführungsweges $p(\xi)$, $\xi \in D_x\}$.

Sind alle Mengen $F_k(x,y),(x,y) \in D_x \times D_y$, eines auswählenden Programmes endlich und alle Abbildungen sowie Prädikate berechenbar[1], so kann dessen Ausführung von einem Rechner übernommen werden, falls dieser so programmiert ist, daß er die nichtdeterministischen Strukturen durch "backtracking"[2]

---

[1] Ein Prädikat p(x), x Element eines Definitionsbereichs D heißt berechenbar, wenn D eine Menge von Worten über einem endlichen Alphabet ist, und eine berechenbare Abbildung f von D in $\mathbb{N}$ existiert, so daß $((p(x) \Leftrightarrow f(x) = 1)$ $(\neg p(x) \Leftrightarrow f(x) = 0))$ gilt; zum Begriff berechenbar siehe (BRAUER-INDERMARK,1968).

[2] Zum Begriff "backtracking" vgl. (RECHENBERG,1975). Zum mit "backtracking" verbundenen Aufwand siehe Fußnote 4, p.130.

auflösen kann[1].

3.3.2. Ein Zusammenhang interaktiver Prozeduren mit auswählenden Programmen

Sei $T = (Q,Q_U,A,d,q_O)$ eine interaktive Prozedur; T erfülle ohne Beschränkung der Allgemeinheit die Voraussetzung, daß genau ein Zustand $q_F \in Q$ mit $B_Q(q_F) = \emptyset$ existiert, in dem alle Interaktionen enden.

Ist $t = \{(q^{(i)},w^{(i)},a(w^{(i)}))\}_{i \in I}$, $I = \{0,i_o,\ldots\}$ ein Lösungsprozeß in T so sei $t_U := \{(q^{(k)},w^{(k)},a(w^{(k)}))\}_{k \in K}$, $K = \{0,k_1,\ldots\} \subset I$ die Teilfolge aller Konfigurationen aus t, die einen Unterbrechungszustand oder q_F enthalten. Dabei seien I und K ohne Einschränkung so, daß gilt: Es ist $|k_s - k_j| \geq 3$ für alle s und j, und ist $k_j = i_l$, so sind $i_{l-1} = k_j-1$ und $i_{l+1} = k_j+1$ (falls $i_{l+1} \in I$). Dann gilt:

Satz 3.3.2.1.: Zu jeder interaktiven Prozedur $T=(Q,Q_U,A,d,q_O)$ existiert ein auswählendes Programm AP_T mit Variablen x, y und z und Definitionsbereichen $D_x = D_y = A^+ \times \mathbb{N}$ und $D_z = A^+$, das erfüllt:

---

[1] Für dieses Auflösen eines nichtdeterministischen in ein deterministisches "Backtrack-Programm" siehe (FLOYD,1967a). Für eine nichtdeterministische Erweiterung einer Programmiersprache siehe die nichtdeterministische Erweiterung von FORTRAN in (COHEN-CARTON,1974) (auf der Grundlage von (FLOYD,1967a)).

(i) ist $t = \{(q^{(i)}, w^{(i)}, a(w^{(i)}))\}_{i \in I}$, $I=\{0,i_0,..,i_n\}$ eine Interaktion in T und ist $t_U = \{(q^{(k)}, w^{(k)}, a(w^{(k)}))\}_{k \in K}$, $K=\{0,k_1,k_2..,k_n\}$, so existieren ein $\xi \in D_x$ und ein Ausführungsweg $p(\xi)$ in $AP_T(\xi)$ der Gestalt $p(\xi) = \{(\xi^{(l)}, \eta^{(l)}) := ((\xi_1^{(l)}, \xi_2^{(l)}), (\eta_1^{(l)}, \eta_2^{(l)}))\}_{l \in L}$ [1], $L=\{k_0, k_1-1, k_1, k_1+1, \ldots\}$ mit:

$$(\xi^{(l)}, \eta^{(l)}) = \begin{cases} (w^{(0)}, a(w^{(0)}, w^{(0)}, a(w^{(0)})), \text{ falls } l=0; \\ (\xi_1^{k_{i-1}}, \xi_2^{k_{i-1}}, w^{(l)}, a(W^{(l)})), \text{ falls } l=k_1-1, \\ k_2-1, \ldots; \\ (w', a, w^{(l)}, a(w^{(l)})), \; w' \in A', \text{ falls} \\ d(q^{(l)}, a(w^{(l)})) \ni \{(q', b, A')\}^{2)}, \; l=k_1, k_2, \ldots, \\ l \neq k_n, \; a \in A; \\ (\xi_1^{(l-1)}, \xi_2^{(l-1)}, w^{(l)}, a(w^{(l)})), \text{ falls} \\ l=k_1+1, k_2+1, \ldots, \; l \neq k_n+1; \\ (\xi_1^{(k_{n-1})}, \xi_2^{(k_{n-1})}, w^{(n)}, a(w^{(n)})), \text{ falls } l=k_n; \end{cases}$$

und Resultat $w^{(n)}$.

---

[1] Die Klammern um die Komponenten der Variablen x und y werden im folgenden weggelassen.

[2] q' und b sind dabei durch t eindeutig bestimmt.

(ii) ist $p(\xi) = \{(\xi_1^{(l)}, \xi_2^{(l)}, \eta_1^{(l)}, \eta_2^{(l)})\}_{l \in L}$,
$L=\{0, l_1-1, l_1, l_1+1, \ldots, l_n\}$[1] ein Ausführungsweg mit Resultat η, $\xi \in D_x$, so existiert eine Interaktion $t=\{(q^{(i)}, w^{(i)}, a(w^{(i)}))\}_{i \in I}$, $I=\{0, \ldots, l_1-1, l_1, l_1+1, \ldots, l_2-1, l_2, l_2+1, \ldots, l_n\}$, die erfüllt:

1. $w^{(o)} = \xi_1^{(o)}$, $a(w^{(o)}) = \xi_2^{(o)}$[2] ;
2. für $i=l_j-1$, $i=l_j$ und $i=l_j+1$ gilt $w^{(i)}=\eta_1^{(i)}$, $a(w^{(i)})=\eta_2^{(i)}$, $j = 1, \ldots, n-1$;
3. für $i=l_j$ ist $w^{(i)} = \xi_1^{(i)}$, $j=1, \ldots, n-1$;
4. $w^{(l_n)} = \eta_1^{(l_n)}$.

Der Satz besagt, daß es zu einer interaktiven Prozedur T ein auswählendes Programm AP_T derart gibt, daß gilt:

a. zu jeder Interaktion in T mit Endinformation (w,a(w)) gibt es einen Ausführungsweg in AP_T mit Resultat w und umgekehrt.
b. die Folgen von Konfigurationen an Start-, Unterbrechungs- und deren direkten Folgezuständen sowie am Endzustand q_F von Interaktionen in T und die Ausführungswege in AP_T "entsprechen" einander.

Der Satz läßt damit den Schluß zu, interaktive Prozeduren mit auswählenden Programmen zu identifizieren, wenn nur die Funktion interner Rechnerkomponenten, nicht aber ihre Implementierung als Turingprogramme betrachtet wird.

---

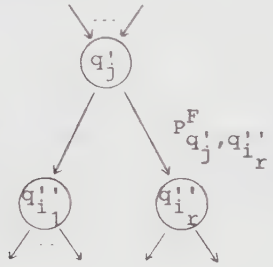
[1] Es gelte ohne Einschränkung $|l_i - l_j| \geq 3$ für alle i und j.

[2] Man beachte, daß zwischen dem Index der aktuellen Stelle und der Stelle selbst hier im allgemeinen nicht unterschieden wird.

Dies wird auch im folgenden Beweis von Satz 3.3.1. deutlich.

Beweis: Zur Konstruktion eines auswählenden Programmes AP_T zu T seien zunächst $D_x = D_y := A^+ \times \mathbb{N}$, $D_z := A^+$; $x=(x_1,x_2)$ und $y=(y_1,y_2)$. Es werden dann die Grundkomponenten von T in Komponenten eines auswählenden Programmes übersetzt. Dabei sind offenbar zwei Fälle zu unterscheiden:

1.

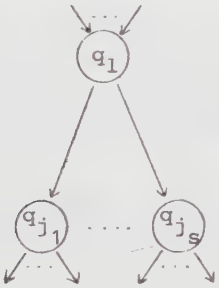


$q'_j \in Q'$ (insbesondere ist hier der Fall $q'_j=q_o$ eingeschlossen);

$q''_{i_1},\ldots,q''_{i_r} \in Q''$[1];

hierbei kann höchstens ein $q_{i_k}=q_F$ sein;

2.



$q_1 \in Q_U$

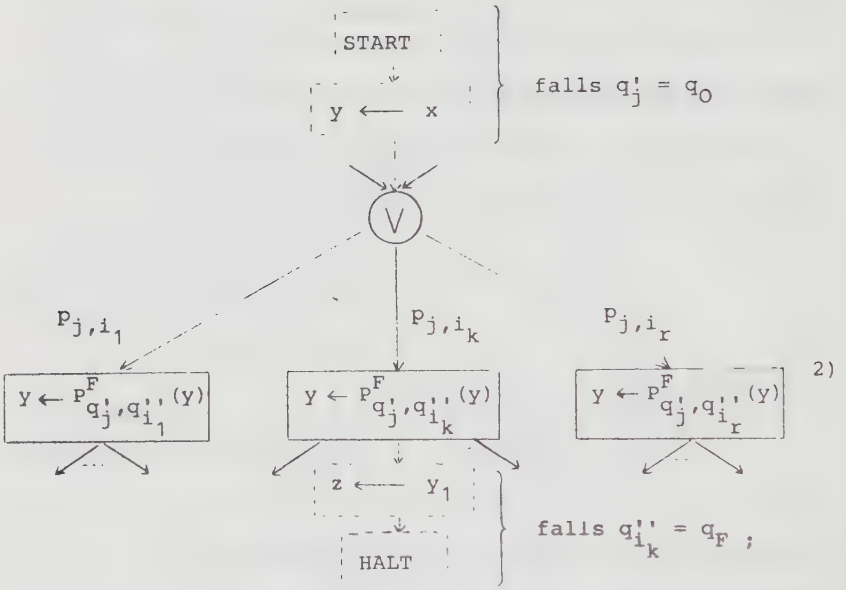
$\{q_{j_1},\ldots,q_{j_s}\}=N(q_1)$[2] $\subset Q'$

Im ersten Fall definiert man nun eine Komponente eines auswählenden Programmes wie folgt:

---

[1] Zur Definition von Q',Q'' und $P^F_{q',q''}$ vgl. 3.2.1.

[2] Zu N(q) vgl. 3.2.1.

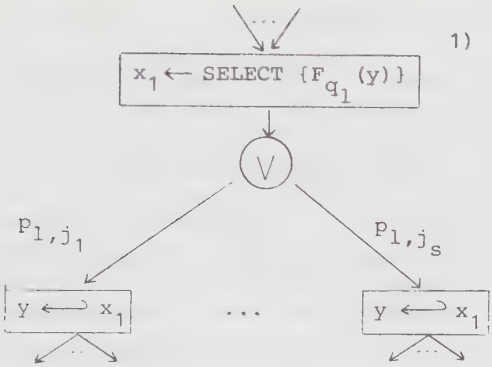


dabei sei $p_{j,i_k} := \begin{cases} T, \text{ falls } y \in DEF\ (P_{q'_j,q''_{i_k}}) \\ F, \text{ falls } y \in D_y\text{-}DEF(P_{q'_j,q''_{i_k}}) \end{cases}$, k=1,..,s .

Im zweiten Fall definiert man eine Komponente eines auswählenden Programmes wie folgt:

---

1) Zuordnungsanweisungen sind Spezialfälle von SELECT-Anweisungen, vgl. 3.3.1.



dabei seien $F_{q_1}(y) := A_{j_1} \cup \ldots \cup A_{j_s}$, falls
$d(q_1, y_z) = \{(q_{j_1}, b_{j_1}, A_{j_1}), \ldots, (q_{j_s}, b_{j_s}, A_{j_s})\}$,

$$p_{1,j_k} := \begin{cases} T, \text{ falls } x_1 \in A_{j_k} \\ F, \text{ falls } x_1 \in A^+ - A_{j_k} \end{cases} , \quad k=1,\ldots,s ;$$

$y \hookleftarrow w$, $w \in A^+$, bezeichne die Einsetzung von w in y_1 an der Stelle y_2; danach enthält y_1 das Wort w und y_2 ist die neue, sich aus dem entsprechenden Zustandsübergang in T ergebende aktuelle Stelle.

Die beiden Aussagen (i) und (ii) des Satzes folgen dann unmittelbar aus der Konstruktion von AP_T.

q.e.d.

---

1) Die Zuordnung an x_1 entspricht der Tatsache, daß die Wahl eines Wertes durch einen Benutzer eine Eingabe darstellt.

Satz 3.3.2.1. erlaubt einige Anmerkungen:

1. AP_T erzeugt alle endlichen Lösungsprozesse in T in ihren zu Beginn, an Unterbrechungen und an ihrem Ende sichtbaren Ausprägungen. Alle möglichen Ausführungswege in AP "umschließen den Zustandsraum"[1] "eventueller Lösungen"[2]; in ihm findet das Suchen von Lösungen des Problems statt, für das T eine Problemlösung angibt. Für einen Benutzer, der nicht am Aussehen der internen Rechnerkomponenten von T interessiert ist, stellt sich dann ein Lösungsprozeß in T im wesentlichen als eine Folge in diesem Zustandsraum dar, die er auf das Erstellen einer Lösung hin zielgerichtet[3] mit einem Rechner erarbeitet.

2. Betrachtet man AP_T statt T als Modell für interaktive Problemlösungen, so kann von Forderung 2.2.(vi) für die internen Rechnerkomponenten abgegangen werden; allerdings ist dann der gewünschte Ablauf der Lösungsprozesse durch geeignete Synchronisationsmaßnahmen sicherzustellen.

---

[1] Vgl. (NILSSON,1971),p.24; die einzelnen Zustände eines derartigen Zustandsraumes sind mögliche Werte für die Variablen von AP_T in Ausführungswegen in AP_T.

[2] Vgl. 1.1.

[3] Vgl. 1.1.; man beachte, daß das "zielgerichtete" Arbeiten eines Benutzers eine Erwartung darstellt; vgl. auch die Modelldiskussion in 2.3.1.

3. AP_T ermöglicht es, die Korrektheit endlicher Lösungsprozesse in T auf der Basis der Korrektheit nichtdeterministischer Programme, vgl. (MANNA,1970), zu betrachten; insbesondere wird dabei der Aktivitätenspielraum eines Benutzers mit einbezogen. Korrektheitsbetrachtungen für interaktive Prozeduren auf der Basis sog. "auswählender Programme" werden nun im folgenden Teil durchgeführt.

3.4. Zur Korrektheit interaktiver Problemlösungen

Korrektheit eines Programmes[1] liegt dann vor, wenn ein Rechner dieses Programm so ausführt, daß dabei in Abhängigkeit von den Eingabedaten die gewünschten Ausgabedaten bestimmt werden[2]. Üblicherweise wird dies durch empirisches Testen von Programmen nachgeprüft. Nun kann aber "das Prüfen von Programmen durch empirisches Testen (kann) höchstens die Gegenwart von Fehlern aufzeigen, niemals jedoch die Korrektheit eines Programmes garantieren"[3]; diese Misere kann mit der auf (FLOYD,1967b), (NAUR,1966) zurückgehenden Methode der Programmverifikation[4] bezüglich zum Programm hinzugefügter "Zusicherungen" überwunden werden[5]. Zusicherungen sind dabei Prädikate, mit denen einzelne Pfeile des Programmflußdiagrammes[6] markiert werden; diese Markierung geschieht

---

[1] Ein Programm sei ein auswählendes Programm, in dem alle SELECT-Anweisungen "übliche" Zuordnungsanweisungen und alle $\bigvee$-BRANCH-Anweisungen Verzweigungen in zwei Programmäste sind.

[2] Vgl. (MANNA,1969), p.119:" A program is said to be correct if its execution terminates and yields the desired result".

[3] Vgl. (WIRTH,1972), p.25.

[4] Synonym: "Program proving".

[5] Für einen Überblick über das Gebiet der Programmverifikation siehe (LINDEN,1972).

[6] Die hier betrachteten Programme liegen in Flußdiagrammform vor, vgl. Fußnote 1.; die Methode ist jedoch auch auf rekursive Programmstrukturen erweiterbar, vgl. (deBAKKER-MEERTENS,1974).

dabei so, daß je eine Zusicherung zu Beginn und am Ende[1] stehen, jede Programmschleife (im Flußdiagramm) von mindestens einer Zusicherung "geschnitten" wird, und jede Programmanweisung zwischen mindestens zwei Zusicherungen liegt. Für jeden Weg w von einer Zusicherung ("Vorbedingung") zu einer weiteren verschiedenen ("Konsequenz") im Flußdiagramm des Programmes, der keine weitere Zusicherung enthält, ist dann die logische Richtigkeit der sogenannten Verifikationsbedingung nachzuweisen[2]:

((Vorbedingung ∧ "Programmfunktion entlang w")

⇒ Konsequenz).

Aus der logischen Richtigkeit aller dieser endlich vielen Verifikationsbedingungen kann man dann folgern: für alle Eingabewerte, die die Vorbedingung am Programmbeginn erfüllen und vom Programm in endlich vielen Schritten zu einem Ausgabewert verarbeitet werden, erfüllt dieser Ausgabewert die Konsequenz am Programmende[3]. Das Programm heißt dann korrekt bezüglich der Eingabe- und Ausgabezusicherung. Es ist klar, daß der Wert einer derartigen Korrektheitsaussage stark von diesen beiden Zusicherungen abhängt.

---

[1] Ohne Beschränkung der Allgemeinheit werde für ein Programm angenommen, daß es genau eine HALT-Anweisung enthält.

[2] Für eine ausführliche methodische Betrachtung siehe (WIRTH,1972), p.24ff., (ELSPAS-GREEN-LEVITT-WALDINGER, 1972).

[3] Vgl. (deBAKKER-MEERTENS,1974), p.6ff.

Offenbar liegen Schwierigkeiten der Methode der Programmverifikation im Aufstellen der Zusicherungen sowie den Nachweisen der logischen Richtigkeit der Verifikationsbedingungen[1], so daß "they (proofs of correctness, d.Verf.) are still limited to small programs, but it is now conceivable to attempt to prove the correctness of small critical modules of a large program"[2]. Eine Möglichkeit zur Weiterentwicklung der Programmverifikation besteht nicht zuletzt in der Einbeziehung interaktiven Problemlösens[3].

Korrektheitsbetrachtungen sind nun allerdings nicht an deterministische Programme gebunden, sondern auch auf nichtdeterministische - und damit auch auf auswählende - Programme erweiterbar.

3.4.1. Korrektheit auswählender Programme

Seien AP ein auswählendes Programm[4], $\phi(x)$ ein totales Prädikat über D_x und $\psi(x,z)$ ein totales Prädikat über $D_x \times D_z$; sei $\xi \epsilon D_x$.

---

1) Es ist dabei auch möglich, daß eine Programmverifikation durch Fehler in den Nachweisen der logischen Richtigkeit der Verifikationsbedingungen verfälscht wird.

2) Vgl. (LINDEN,1972), p.201; dort werden auch die Schwierigkeiten der Methode sowie Möglichkeiten zu ihrer Überwindung diskutiert.

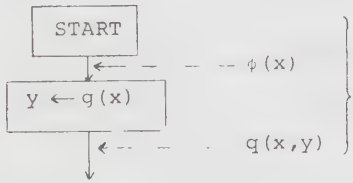
3) Vgl. etwa (ELSPAS-GREEN-LEVITT-WALDINGER,1972), (FLOYD, 1972).

4) Vgl. Definition 3.3.1.1.

Definition 3.4.1.1.: AP heißt partiell korrekt bezüglich ϕ und ψ genau dann, wenn für alle $\xi \in D_x$ und alle $\eta \in D_z$ folgt, daß die Gültigkeit von $\phi(\xi)$ und $\eta \in AP^F(\xi)$ implizieren, daß $\psi(\xi,\eta)$ gilt[1].

Korrektheit auswählender Programme läßt sich nun analog zu der deterministischer Programme nachweisen[2]. Es sei dazu wieder vorausgesetzt, daß AP so mit Zusicherungen markiert ist, daß je eine zu Beginn und Ende steht, jede Schleife in AP von mindestens einer Zusicherung "geschnitten" wird, und jede Anweisung zwischen zwei Zusicherungen liegt[3]. Für das Aufstellen der Verifikationsbedingungen ist dann offenbar die Betrachtung der folgenden Fälle wesentlich:

a.



ϕ ist ein totales Prädikat über D_x, $q(x,y)$ ein totales Prädikat über $D_x \times D_y$; die zugehörige Verifikationsbedingung ist:

$\forall x \forall y((\phi(x,y) \wedge (y=g(x)) \Rightarrow q(x,y))$

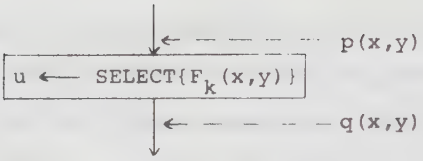
---

[1] Vgl. (MANNA,1970), p.15.

[2] Vgl. (deBAKKER-MEERTENS,1974), p.24,25.

[3] Vgl. (MANNA,1970), p.10, (deBAKKER-MEERTENS,1974), p.24.

b.



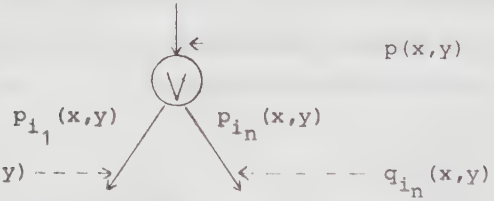
p und q sind totale Prädikate über $D_x \times D_y$; die zugehörige Verifikationsbedingung ist:

$\forall x \forall y \forall x'((p(x,y) \wedge (x' \in F_k(x,y))) \Rightarrow q(x',y))$, falls u=x ,

$\forall x \forall y \forall y'((p(x,y) \wedge (y' \in F_k(x,y))) \Rightarrow q(x,y'))$, falls u=y.

Für u=y und $|F_k(x,y)| \leq 1$ für alle x und y ist die übliche Zuordnungsanweisung von Programmen eingeschlossen.

c.



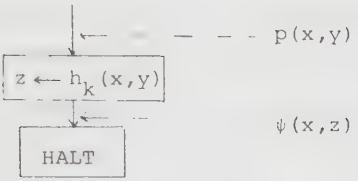
$p, q_{i_1}, \ldots, q_{i_n}$ sind totale Prädikate über $D_x \times D_y$. Die Verifikationsbedingung, um in eine der Verzweigungen $j_1, \ldots, j_l$, $\{j_1, \ldots, j_l\} \subset \{i_1, \ldots, i_n\}$, zu gelangen, ist:

$\forall x \forall y(((p(x,y) \wedge p_{j_1}(x,y)) \Rightarrow q_{j_1}(x,y)) \vee \ldots$

$\ldots \vee ((p(x,y) \wedge p_{j_l}(x,y)) \Rightarrow q_{j_l}(x,y)))$

Für n=2 und $p_{i_1} = \neg p_{i_2}$ ist die übliche Verzweigungsanweisung von Programmen eingeschlossen.

d.

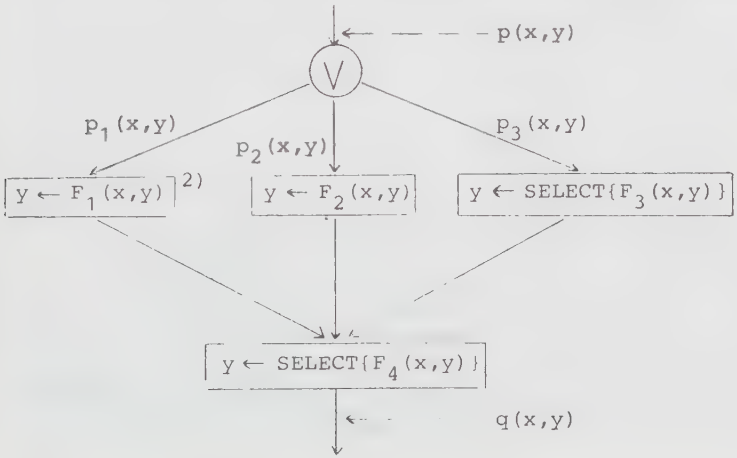


p ist ein totales Prädikat über $D_x \times D_y$, ψ ein totales Prädikat über $D_x \times D_z$; die zugehörige Verifikationsbedingung ist:

$$\forall x \forall y \forall z ((p(x,y) \wedge (z=h_k(x,y))) \Rightarrow \psi(x,z))$$

Das folgende Beispiel verdeutlicht die Verifikation eines auswählenden Programmes[1]:

Beispiel 3.4.1.1.: Es sei AP ein auswählendes Programm, das bereits wie oben beschrieben mit Zusicherungen markiert ist; dabei sei etwa folgender Teil von AP zu verifizieren:



---

[1] Für weitere Beispiele siehe (MANNA,1971), p.249,252 sowie 3.4.3.

[2] Zuordnungsanweisungen sind Spezialfälle von SELECT-Anweisungen.

Dann ist die logische Richtigkeit der folgenden Verifikationsbedingung nachzuweisen:

$$\forall x \forall y \forall y' \forall y'' (((p(x,y) \wedge ((p_1(x,y) \wedge (y'=F_1(x,y))) \vee (p_2(x,y) \wedge (y'=F_2(x,y))) \vee (p_3(x,y) \wedge (y' \in F_3(x,y)))) \wedge (y'' \in F_4(x,y'))) \Rightarrow q(x,y'')) \ ;$$

die Quantifizierung von x erfolgt über D_x, die von y, y' und y'' über D_y.

Hat man für ein mit Zusicherungen markiertes Programm AP alle Verifkationsbedingungen aufgestellt und ihre logische Richtigkeit nachgewiesen, so läßt sich daraus die Korrektheit von AP bezüglich Eingabe-Vorbedingung und Ausgabe-Konsequenz folgern[1].

Auf der Basis der Korrektheit auswählender Programme lassen sich nun Aussagen über die Korrektheit interaktiver Problemlösungen gewinnen.

3.4.2. Korrektheit interaktiver Problemlösungen

Das sich zu Beginn, am Ende und an Unterbrechungen abspielende Geschehen in Lösungsprozessen einer interaktiven Problemlösung kann - vgl. 3.3.2.1. - durch ein auswählendes Programm AP_T dargestellt werden, wenn die Problemlösung

---

[1] Vgl. (deBAKKER-MEERTENS,1974), p.24,25, Theorem 4.1.

durch ein interaktive Prozedur T beschrieben wird. Der Spielraum an Aktivitäten, den ein Benutzer an einer Unterbrechung zur Verfügung hat, ist dabei stets eine Menge $F_k(x,y)$, wenn (x,y) die bis zu dieser Unterbrechung erarbeitete Information ist[1]. Seien nun $\phi(x)$ und $\psi(x,z)$ zwei totale Prädikate derart, daß AP_T korrekt bezüglich ϕ und ψ ist. Dies bedeutet dann: führt ein Benutzer mit einer Startinformation ξ, für die $\phi(\xi)$ gilt, zusammen mit einem Rechner einen interaktiven Lösungsprozeß durch, der ein Resultat η ergibt, so gilt $\psi(\xi,\eta)$.

Beschreibt etwa T eine interaktive Problemlösung eines betrieblichen Entscheidungsproblems[2], das von gewissen gegebenen Informationen ausgehend zur Bestimmung einer Handlung führen soll, und ist AP_T korrekt bezüglich $\phi(x)$ und $\psi(x,z)$, so heißt dies: Liegen Ausgangsdaten vor, die die Anforderungen von ϕ erfüllen, und führen Benutzer und Rechner mit diesen Daten einen Entscheidungsprozeß durch, der einen regulären Abschluß findet (an einer HALT-Anweisung), so gilt: die erarbeitete Handlung erfüllt die in ψ niedergelegten Anforderungen an die Resultate von Entscheidungsprozessen des betrachteten Entscheidungsproblems.

---

[1] Vgl. den Beweis von Satz 3.3.2.1.

[2] Vgl. 1.2.2.1.

Bevor das Nachweisen der Korrektheit interaktiver Problemlösungen noch im Rahmen der interaktiven Transportmitteldisposition aus 2.1. verdeutlicht wird, erscheinen einige Bemerkungen angebracht:

1. Für Nachweise der Korrekheit interaktiver Problemlösungen im hier betrachteten Sinn treten (derzeit) die gleichen Probleme auf, wie bei der Verifikation "üblicher" deterministischer Programme; vgl. die Einführung in 3.4.. Da die Untersuchungen auf dem Gebiet der Korrektheitsbeweise jedoch noch im Fluß sind[1)], läßt sich über eine Lösung dieser Probleme und die damit verbundene Einbeziehung von Korrektheitsbeweisen in die Anwendung noch nichts abschließendes sagen.
2. In den Erörterungen zur Korrektheit von interaktiven Problemlösungen ist die Betrachtung der internen Rechnerkomponenten auf deren Funktion reduziert. Liegen diese Funktionen als Rechnerprogramme vor (in interaktiven Prozeduren sind es Turingprogramme), so kann wiederum deren Korrektheit bezüglich vorgegebener Zusicherungen mit der Methode der Programmverifikation nachgewiesen werden.

[1)] Darunter fallen Untersuchungen zum mehr oder weniger stark rechnerunterstütztem Beweisen der Richtigkeit von Theoremen, über die Generierung von Zusicherungen, die Methodik der Programmverifikation, u.ä.; für entsprechende Hinweise siehe etwa (LINDEN,1972), (ELSPAS-GREEN-LEVITT-WALDINGER,1972).

3. Hat man ein Programm so mit Zusicherungen markiert, daß sich seine Korrektheit bezüglich Eingabe- und Ausgabezusicherung nachweisen läßt, so schließen die Zusicherungen die Semantik des Programmes ein[1]. Dies erlaubt, auf der Ebene von Korrektheitsüberlegungen weitere Eigenschaften von Programmen, wie deren (funktionale) Äquivalenz und das (reguläre) Enden von Lösungsprozessen in ihnen zu betrachten[2]. Für interaktive Problemlösungen sind solche Überlegungen von der Anwendung her gesehen nur für die internen Rechnerkomponenten, nicht aber für die hier betrachtete übergeordnete Korrektheit auf der Basis auswählender Programme von Interesse: da ein Benutzer in der Regel starke Unterbrechungen zur Verfügung hat (vgl. 2.2.(iii)2.), kann er unabhängig von der Problemlösungslogik Lösungsprozesse beenden; da interaktive Problemlösungen häufig einen noch stärkeren Einmaligkeitscharakter als Programme haben, sind auch Äquivalenzüberlegungen, zumindest für beliebige[3] interaktive Problemlösungen, von untergeordneter Bedeutung.

---

[1] Vgl. (FLOYD,1967b).

[2] Vgl. etwa (MANNA,1969), p.125ff., (MANNA,1971), p.250ff.

[3] Äquivalenzüberlegungen sind höchstens in dem Fall von Interesse, wenn eine interaktive Problemlösung verändert werden soll, etwa um sie sowohl an geübte wie ungeübte Benutzer anzupassen; dies kann dadurch geschehen, daß etwa ein geübter Benutzer die Möglichkeit erhält, Lösungsschritte überspringen zu können.

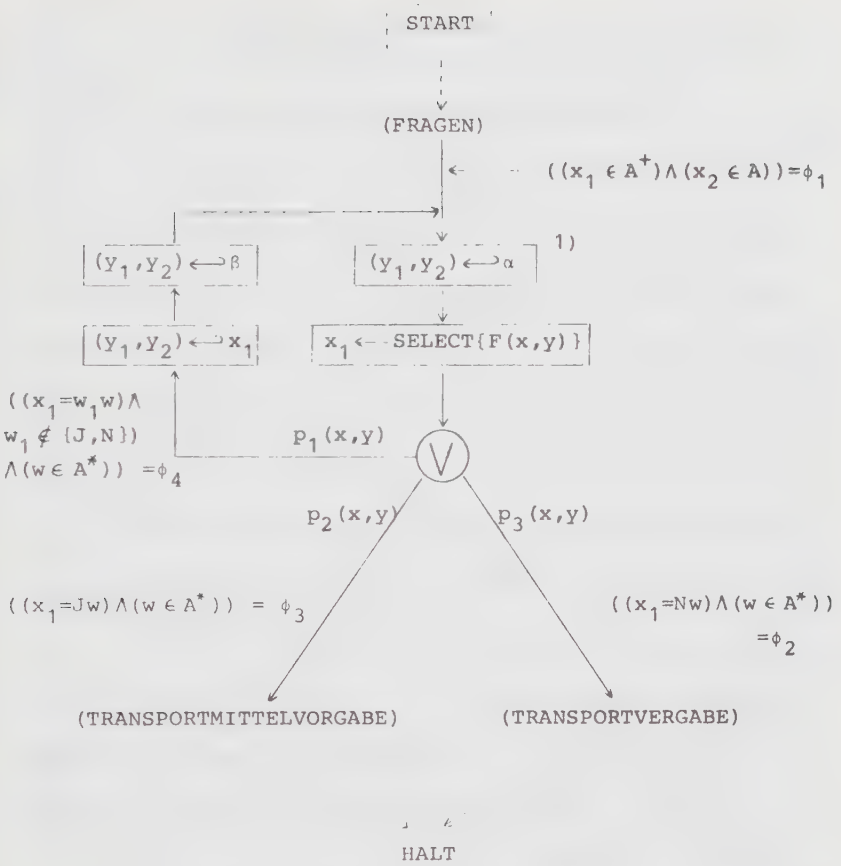
4. Die Methode der Programmverifikation ist nicht auf Programme beschränkt, in denen alle Lösungsprozesse sequentiell ablaufen, sondern ist auch auf Programme mit parallelen, zu synchronisierenden Lösungsprozessen erweiterbar[1].

3.4.3. Korrektheit innerhalb der interaktiven Transportmitteldisposition

Es wird wieder der Teil der Transportmitteldisposition aus 2.3.3. betrachtet. Die interaktive Prozedur T aus 2.3.3. läßt sich als ein auswählendes Programm AP wie folgt darstellen: Seien $D_x = D_y = A^+ \times \mathbb{N}$, $x = (x_1, x_2)$ und $y = (y_1, y_2)$; da T einen Ausschnitt der gesamten Problemlösung darstellt, wird auf explizite START- und HALT-Anweisungen verzichtet und vorausgesetzt, daß den Variablen x und y zu Beginn der betrachteten Problemlösungskomponente bereits Werte aus $A^+ \times \mathbb{N}$ zugewiesen sind:

---

[1] Vgl. (LEVITT,1972); damit lassen sich Korrektheitsbetrachtungen auch auf interaktive Problemlösungen ausdehnen, die rechnerseitig parallele Prozesse beinhalten, wie etwa die in (DENERT,1975) beschriebene Implementierung.



Figur 3.4.3.1.

Dabei seien für $(x,y) \in D_x \times D_y$:

$F(x,y) := \{w \in A^+ : \perp(w) \leq m\}$;

---

[1] Zur Definition von " $\hookleftarrow$ " vgl. den Beweis von Satz 3.3.2.1.

$$p_1(x,y) := \begin{cases} T, \text{ falls } x_1 \neq J,N,\ l(x_1) \leq m, \\ F \text{ sonst;} \end{cases}$$

$$p_2(x,y) := \begin{cases} T, \text{ falls } x_1 = Jw,\ w \in A^*,\ l(x_1) \leq m, \\ F \text{ sonst,} \end{cases}$$

$$p_3(x,y) := \begin{cases} T, \text{ falls } x_1 = Nw,\ w \in A^*,\ l(x_1) \leq m, \\ F \text{ sonst.} \end{cases}$$

AP ist mit vier Zusicherungen $\phi_1,..,\phi_4$ so markiert, daß die Schleife "geschnitten" wird, und jede Anweisung auf einem Weg im Flußdiagramm zwischen zwei Zusicherungen liegt. Die Zusicherungen spiegeln den Textverarbeitungscharakter von AP wider. Will man die Korrektheit von AP in Bezug auf jeden einzelnen Ausgang der $\bigvee$-BRANCH-Anweisung untersuchen, so ist die logische Richtigkeit von fünf Verifikationsbedingungen nachzuweisen. Hier soll nun beispielhaft nur die Verifikationsbedingung mit ϕ_1 als Vorbedingung und ϕ_2 als Konsequenz betrachtet werden; es ist dabei zu beachten, daß AP in der Schleife von der Zusicherung ϕ_4 "(auf)geschnitten" wird, so daß die logische Richtigkeit des folgenden Ausdrucks nachzuweisen ist:

$$\bigvee x \bigvee x' \bigvee y(((\phi_1(x,y) \wedge (x_1' \leftarrow \text{SELECT}\{F(x,((x_1,x_2) \hookleftarrow \alpha)\}) \\ \wedge p_3(x',y)) \Rightarrow \phi_2(x',y)) \ .$$

Setzt man ϕ_1, ϕ_2, F und p_3 ein, so folgt sofort die logische Richtigkeit dieses Ausdrucks; sie besagt:

Wählt ein Benutzer in einem Lösungsprozeß auf die Aufforderung "WEITERE TRANSPORTMITTELVERGABE GEWUENSCHT?" als Antwort ein beliebiges mit N beginnendes Wort einer Länge kleiner oder gleich m, so verzweigt der Lösungsprozeß mit Sicherheit[1] zu der Marke TRANSPORTVERGABE; "übliches" Testen würde diese Aussage mit Sicherheit im wesentlichen nur für alle, in den Tests benutzten, mit N beginnenden Antworten zulassen.

---

[1] Diese Sicherheit gilt stets vorbehaltlich von Fehlern in der Verifikation.

3.5. Zur Komplexität interaktiver Problemlösungen

Mit Hilfe interaktiver Prozeduren ist es möglich, Komplexität[1] interaktiver Problemlösungen zu betrachten; hier soll dabei die für Anwendungen interessantere Zeitkomplexität[2] - im folgenden einfach als Komplexität bezeichnet - erörtert werden.

Man überlegt sich zunächst leicht, daß eine interaktive Prozedur $T=(Q,Q_U,A,d,q_0)$ als Beschreibung einer nichtdeterministischen Turingmaschine[3] aufgefaßt werden kann[4]. Damit können die Zeitkomplexität (sowie weitere Komplexitätsmaße[5]) einer interaktiven Prozedur definiert werden[6]. Mit ihr hat man ein Komplexitätsmaß für interaktive Problemlösungen, das jedoch alle Möglichkeiten einschließt, die ein Benutzer an Unterbrechungen zur Verfügung hat, ohne die in einem

---

1) Für einen Überblick über Komplexitätstheorie siehe etwa (HARTMANIS-HOPCROFT,1971), (HOPCROFT,1974).

2) Zur Definition der Zeitkomplexität einer deterministischen Turingmaschine siehe (AHO-HOPCROFT-ULLMAN,1974), p.30; für nichtdeterministische Turingmaschinen (AHO-HOPCROFT-ULLMAN,1974), p.367; für Zeitkomplexität betrachtet im allgemeineren Rahmen von Komplexitätsmaßen siehe etwa (BORODIN,1973), p.37ff, ("dynamic resource functions").

3) Zur Definition einer (k-bändrigen) nichtdeterministischen Turingmaschine siehe (AHO-HOPCROFT-ULLMAN,1974), p.365.

4) Vgl. (ROTH,1975a) für eine Darstellung von T als dreibändrige nichtdeterministische Turingmaschine.

5) Zum Begriff siehe (HARTMANIS-HOPCROFT,1971), p.447/448.

6) Vgl. Fußnote 1.

Lösungsprozeß erbrachte "intellektuelle Leistung" eines Benutzers deutlich zu machen. Diese wird nun aber bei interaktivem Problemlösen u.a. dann benötigt, wenn die Komplexität von Algorithmen zu lösender Probleme[1] bereits für in der Anwendung interessante Problemgrößen "zu groß" ist - etwa nicht von polynomialer Zeit[2]; man hat dann die Hoffnung, daß die Mitwirkung eines Benutzers die benötigte Zeit zumindest in den anwendungsmäßig interessanten Fällen auf eine möglichst polynomiale Abhängigkeit von der Problemgröße nicht zu großen Grades herabdrückt[3]. Dies legt nahe, bei der Betrachtung der Komplexität einer interaktiven Problemlösung deren gesamte Komplexität unter dem Gesichtspunkt anzusehen, daß die Komplexität von Lösungsprozessen wesentlich durch das Verhalten eines Benutzers bestimmt wird;

---

[1] Die Existenz solcher Algorithmen ist dabei hier vorauszusetzen; sie ist nicht stets gegeben, etwa dann nicht, wenn ein Benutzer in einem betrieblichen Entscheidungsproblem (vgl. 1.2.2.1.) eine subjektive Datenbewertung durchführt; vgl. die Diskussion von (LODISH,1971) in 1.2.2.2.

[2] Ist $T(n)$, n Länge eines Eingabewortes, die Zeitkomplexität einer deterministischen Turingmaschine T, so "arbeitet" T in polynomialer Zeit, wenn ein Polynom $f(x)$ existiert, so daß $T(n) \leq f(n)$ ist für alle Eingabeworte der Länge n, bei denen T nach endlich vielen Schritten (Konfigurationen) stehen bleibt; damit ist auch die Art des verwendeten Zeitbegriffs klar.

[3] Vgl. (GRIESE-MÜHLBACHER,1976), Teil 5.; ein Beispiel für derartiges Vorgehen wird in (KROLAK-FELTS-MARBLE,1971) beschrieben; dort werden interaktiv Lösungen (allerdings in der Regel nur suboptimale) des (hier weiter unten ausführlicher behandelten) Travelling-Salesman-Problems bestimmt.

m.a.W. ist es also von Interesse, die Komplexität von Lösungsprozessen in einer interaktiven Problemlösung relativ zur in die Kommunikation eingebrachten Leistung eines Benutzers zu betrachten. Wie dies auf der Basis interaktiver Prozeduren möglich ist, soll die folgende Beispielüberlegung verdeutlichen.

Gesucht seien Lösungen des "Travelling-Salesman-Problems", d.h., gesucht sind kostenminimale Kreise (Rundtouren) in einem mit reellen Kosten bewerteten einfachen Graphen $G = (V,R)$[1]. Über Algorithmen zur Lösung des Travelling-Salesman-Problems ist bekannt, daß sie zur Klasse der sogenannten "NP-vollständigen" Algorithmen gehören, und damit im Fall einer negativen Lösung des "NP = P - Problems"[3] nicht von einer in polynomialer Zeit arbeitenden deterministischen Turingmaschine dargestellt werden können[4]. Sei nun T eine interaktive Prozedur, zunächst ohne Unterbrechungszustände, die den bekannten Algorithmus von Little[5] zur

---

[1] Vgl. (NOLTEMEIER,1976), Teil 10.2.

[2] Vgl. (AHO-HOPCROFT-ULLMAN,1974); speziell zur Definition von "NP-complete" p.373.

[3] Zum "P = NP -Problem" - es ist die bislang ungelöste Frage, ob jede in polynomialer Zeit arbeitende nichtdeterministische Turingmaschine durch eine äquivalente deterministische in polynomialer Zeit simuliert werden kann - siehe etwa (HOPCROFT,1974), p.623.

[4] Eine Simulation einer nichtdeterministischen durch eine deterministische Turingmaschine durch Enumeration aller Möglichkeiten erfordert im allgemeinen exponentiellen Aufwand, vgl. (AHO-HOPCROFT-ULLMAN,1974), p.367; dies gilt also auch für das in 3.3.1. angesprochene "backtracking".

[5] Siehe dazu etwa (NOLTEMEIER,1976), p.180ff.; daß dieser Algorithmus "nicht sonderlich effizient" ((NOLTEMEIER,1976), p.180) ist, spielt für die folgenden Überlegungen keine Rolle.

Bestimmung von Lösungen des Travelling-Salesman-Problems beschreibt. Dann ist aufgrund des "backtracking" [1] erfordernden "Branch-and-Bound" Prinzips[2] des Algorithmus klar, daß T mit nichtpolynomialer Komplexität T(n) arbeitet[3], n Maß für die Problemgröße, etwa $n = |V|$. T werde nun durch einen Unterbrechungszustand wie folgt erweitert: An der Stelle des Algorithmus, an der der Rechner eine Ecke im Lösungsbaum [4] festgelegt hat (d.h. die Zugehörigkeit oder Nichtzugehörigkeit eines Pfeiles aus R zu einer Rundtour), erreicht der Rechner einen Unterbrechungszustand; in ihm "fragt" der Rechner, ob die in G bis zu dieser Ecke im Lösungsbaum bestimmtem Pfeile Elemente einer optimalen Rundtour sind. Antwortet der Benutzer mit Ja, so fährt der Rechner dann an der festgelegten Ecke mit dem Algorithmus fort; antwortet er nicht mit Ja, so wird der Lösungsprozeß an der komplementären[5] Ecke des Lösungsbaumes fortgesetzt. Man sieht sofort, daß die Effizienz dieses modifizierten

---

[1] Vgl. 3.3.1.

[2] Siehe dazu allgemein (NOLTEMEIER,1976), p.177-179.

[3] Vgl. auch (AHO-HOPCROFT-ULLMAN,1974), Theorem 10.9., p.389.

[4] Zur Konstruktion dieses Lösungsbaumes siehe (NOLTEMEIER, 1976), p.180ff.

[5] Ist eine Ecke im Lösungsbaum durch die Zugehörigkeit eines Pfeils aus R zu einer Rundtour gekennzeichnet, so sei ihre komplementäre Ecke durch die Nichtzugehörigkeit dieses Pfeils gegeben.

Verfahrens von Little wesentlich durch die Leistung eines Benutzers bestimmt wird; zwei Grenzfälle können dabei unterschieden werden:

1. ein Benutzer ist in der Lage, wie das Orakel einer Turingmaschine mit Orakel[1] stets "richtig" zu antworten; d.h. hier, daß er bei einer Frage des Rechners stets weiß, ob der gerade betrachtete Pfeil zu einer optimalen Rundtour gehört oder nicht; interaktives Problemlösen in T kann dann wie das Arbeiten einer Turingmaschine mit Orakel aufgefaßt werden;
2. ein Benutzer antwortet relativ willkürlich; hier etwa, weil seine Kenntnisse über einen zugrundeliegenden Graphen oder die verwendete Lösungsmethode gering sind, und die Rechnerunterstützung nur mangelhaft ist; in diesem Fall ist die in interaktiven Prozeduren getroffene Beschreibungsannahme - nichtdeterministisches Auswählen, vgl. 2.2.(iv) - besonders ausgeprägt erfüllt.

In der Regel wird das Verhalten eines Benutzers zwischen diesen beiden Extremen liegen[2].

---

[1] Zur Definition einer "Turingmaschine mit Orakel" siehe (ROGERS,1967), p.129ff.

[2] Dies widerspricht nicht der im Modell interaktive Prozedur gemachten Annahme über das Verhalten eines Benutzers; vgl. auch 2.2.(iv).

Handelt ein Benutzer stets wie in 1. beschrieben, so ist jeder Lösungsprozeß in T zur Bestimmung einer optimalen Rundtour in einem Graphen G = (V,R) offenbar mit f(|V|) Schritten, f Polynom, führbar ("der Spielraum der Benutzeraktivitäten wird nicht ausgenutzt"), also in polynomialer Zeit. Handelt ein Benutzer dagegen stets wie in 2. beschrieben, so wird im allgemeinen die große Komplexität von Algorithmen zur Bestimmung optimaler Rundtouren[1] (hier des Little-Algorithmus) zum Tragen kommen ("der gesamte Spielraum des Benutzers an Aktivitäten wird ggf. benutzt"); d.h. es gibt kein Polynom, durch das die Anzahl von Lösungsschritten zur Bestimmung einer optimalen Rundtour in Abhängigkeit von |V|nach oben beschränkt wird. Im Normalfall liegt dann die Anzahl von Schritten in Lösungsprozessen zwischen den beiden Extremen.

Die Beispielüberlegungen machen deutlich:

a. die Komplexität von Lösungsprozessen in einer interaktiven Problemlösung hängt neben der Komplexität implementierter Rechnerkomponenten wesentlich von der in sie eingebrachten Leistung eines Benutzers ab; dabei ist allerdings notwendig, daß diese Leistung entsprechend ihrer Aussagekraft in der Problemlösung geeignet berücksichtigt wird[2];

---

[1] Vgl. (AHO-HOPCROFT-ULLMAN,1974), p.389 (Theorem 10.9.).

[2] Im oben beschriebenen Beispiel einer interaktiven Prozedur zur Bestimmung einer Lösung des Travelling-Salesman-Problems in einem Graphen wird dies dadurch erreicht, daß die Lösungsprozesse durch die Antworten eines Benutzers gesteuert ablaufen.

b. mit Hilfe interaktiver Prozeduren läßt sich die Komplexität von Lösungsprozessen in einer interaktiven Problemlösung relativ zur Leistung eines Benutzers verdeutlichen: die Komplexität hängt davon ab, ob sich die Leistung eines Benutzers mehr der eines Orakels in einer Turingmaschine mit Orakel oder mehr dem nichtdeterministischen Verhalten innerhalb einer nichtdeterministischen Turingmaschine nähert.

4. Schlußbemerkungen

Interaktives Problemlösen beinhaltet die Möglichkeit einer "Symbiose" aus Mensch und Rechner, die die Problemlösungsfähigkeiten beider in erheblichem Maß erweitern kann. Dies ist in der vorliegenden Arbeit im Rahmen des immer mehr Bedeutung erlangenden interaktiven Lösens betrieblicher Entscheidungsprobleme deutlich geworden.

Aus den Fragen, die interaktives Problemlösen aufwirft, ist dann hier das nicht nur von der Anwendung her interessante Problem aufgegriffen worden, Eigenschaften interaktiver Problemlösungen und in ihnen führbarer Lösungsprozesse allgemein zu charakterisieren. Dazu ist das Modell "interaktive Prozedur" aus Beschreibungsmerkmalen interaktiver Problemlösungen abgeleitet worden.

Auf seiner Basis ist dann der Versuch unternommen worden, Struktur interaktiver Problemlösungen sowie Korrektheit und Komplexität interaktiver Lösungsprozesse in allgemeinem Rahmen darzustellen; auf diese Weise sind allgemeine Aussagen über Entwurf, Implementierung und Benutzung interaktiver Problemlösungen gewonnen worden.

5. Literatur

Aho, A.V. (Hrsg.), Currents in the Theory of Computing, Prentice Hall, Englewood Cliffs, 1973;

Aho, A.V., Hopcroft, J.E., Ullman, J.D., The Design and Analysis of Computer Algorithms, Addison Wesley, Reading, Mass., 1974;

Angel, E.S., Bekey, G.A., Adaptive finite-state models of manual control systems, IEEE Transactions on Man-Machine Systems 9(1968), p.15-20;

Ansoff, H.J., Management-Strategie, Verlag Moderne Industrie, München, 1966;

Bakker, de, J.W., Meertens, L.G., On the completeness of the inductive assertion method, Vorlesungsmanuskript, Advanced Course on Semantics of Programming Languages, Saarbrücken, 1974;

Bamberg, G., Coenenberg, A.G., Betriebswirtschaftliche Entscheidungslehre, Verlag Vahlen, München,1974;

Benayoun, R., Tergny, J., Critères multiples en programmation mathématique: Une solution dans le cas linéaire, Revue Francaise d'Informatique et de Recherche Opérationelle 3(1969), Serie V, p.31-56;

Bloech, J., Ihde, G.-B., Betriebliche Distributionsplanung, Physica, Würzburg, 1972;

Booth, T.L., Sequential Machines and Automata Theory, John Wiley, New York, 1971;

Bormann, P.R., Grimm, E.H., An analysis of selected interactive systems, in: (SHERWOOD,1972), 1972, p.315-388;

Borodin, A., Computational complexity: theory and practice, in (AHO,1973), 1 73, p.35-89;

Brady, R.H., Computers in top-level decision making, Harvard Business Review 45(1967), p.67-76;

Brauer, W., Indermark, K., Algorithmen, rekursive Funktionen und formale Sprachen, Bibliographisches Institut, Mannheim,1968;

Brinch Hansen, P., Operating System Principles, Prentice Hall, Englewood Cliffs,1973;

Brown, J.S., Burton, R.R., Zdybel, F., A model-driven question-answering system for mixed-initiative computer-assisted instruction, IEEE Transactions on Systems, Man and Cybernetics 3(1973), p.248-257;

Brown, T., Kierer, M., The effect of language design on time-sharing operational efficiency, International Journal of Man-Machine Studies 7(1975), p.233-247;

Brunnstein, K., Haefner, K., Händler, W., (Hrsg.), Rechner-Gestützter Unterricht, Lecture Notes in Computer Science 17(1974), Springer, Berlin, 1974;

Campbell-Kelley, M., An Introduction to Macros, Elsevier, Amsterdam, 1973;

Carbonell, J.R., On man-computer interaction: a model and some related issues, IEEE Transactions on Systems Science and Cybernetics 5(1969), p.16-26;

Carter, E., Cohen, K., Portfolio aspects of strategic planning, in: (GROCHLA-SZYPERSKI,1973), 1973, p.497-531;

Cohen, J., Carton, E., Non-deterministic FORTRAN, The Computer Journal 17(1974), p.44-51;

Denert, E., Eine Methode für Entwurf und Implementierung interaktiver Software-Systeme dargestellt am Beispiel des PLAN2D-Editors und des ADAGE-Graphics-Systems, Bericht 75-23 der Technischen Universität Berlin, Fachbereich Kybernetik, 1975;

Diers, G., Haberlandt, K., Planen im Dialog mit APL, IBM-Nachrichten 23(1973), Heft 214, P.510-526;

DIN 66001, Sinnbilder für Datenfluß- und Programmablaufpläne, Deutsche Industrienorm (DIN) 66001, Beuth Verlag, Berlin, Oktober 1969;

DIN 44300, Informationsverarbeitung, Begriffe, Deutsche Industrienorm (DIN) 44300, Beuth Verlag, Berlin, März 1972;

Dinkelbach, W., Modell - ein isomorphes Abbild der Wirklichkeit?, in: (GROCHLA-SZYPERSKI,1973), 1973, p.153-161;

Dirlewanger, W., Dobler, K.-U., Hieber, L., Roos, P., Rzehak, H., Schneider, H.-J., Unger, C., Einführung in Teilgebiete der Informatik I, Göschen, Berlin, 1972;

Dworatschek, S., Management-Informations-Systeme, de Gruyter, Berlin,1971;

Eason, K.D., Damodaran, L., Stewart, T.F.M., Interface problems in man-computer interaction, in: (MUMFORD-SACKMAN,1975), p.91-105;

Edström, O., Man-Computer Decision Making, Dissertation, Göteborg, 1973;

Fandel, G., Wilhelm, J., Zur Entscheidungstheorie bei mehrfacher Zielsetzung, Zeitschrift für OR 20(1976), p.1-21;

Emmert, P., INBESY - Planung und Beurteilung von Investitionen in einem Mensch-Maschine-Kommunikations-System, OR-Praxis, Heft 1, 1975, p.51-94;

Elspas, B., Green, M.W., Levitt, K.N., Waldinger, R.J., Research in interactive program proving techniques, Stanford Research Institute, Menlo Park, California, SRI Project 8398, May 1972;

Floyd, R.W., Nondeterministic algorithms, Journal of the ACM 14(1967a), p.636-644;

Floyd, R.W., Assigning meaning to programs, in: Mathematical Aspects of Computer Science, Vol.19 (J.T. Schwartz, Hrsg.), Amer. Math. Soc., Providence, 1967b, p.19-32;

Floyd, R.W., Towards interactive design of correct programs, Proceedings IFIP Congress, Ljubljana, 1971, North-Holland, Amsterdam, 1972;

Gäfgen, G., Theorie der wirtschaftlichen Entscheidungen, Verlag Mohr, Tübingen, 1968;

Gilman, L., Allen, J.R., APL-Anwendungen auf IBM-Systemen, Verlag Oldenbourgh, München,1973;

Glatthaar, W., Problemlösen im Dialog mit dem Computer, Dissertation, Stuttgart, 1974;

Griese, J., Adaptive Verfahren im betrieblichen Entscheidungsprozeß, Physica, Würzburg, 1972;

Griese, J., Mühlbacher, J., Konstruktionsprizipien und Problembereiche für Mensch-Maschinen-Kommunikationssysteme, in: (NOLTEMEIER,1976a);

Grochla, E., Grundprobleme der Wirtschaftlichkeit in automatisierten Datenverarbeitungssystemen, in: (GROCHLA-1970a), 1970, p.15-33;

Grochla, E. (Hrsg.), Die Wirtschaftlichkeit automatisierter Datenverarbeitungssysteme, Verlag Gabler, Wiesbaden, 1970a;

Grochla, E., Analyse gegenwärtiger und zukünftiger Entwicklungstendenzen bei der Planung, Entwicklung und Implementierung computer-gestützter Entscheidungssysteme, in: (GROCHLA,1971a), 1971, p.219-227;

Grochla, E. (Hrsg.), Computergestützte Entscheidungen in Unternehmen, Verlag Gabler, Wiesbaden, 1971a;

Grochla, E., Szyperski, N., (Hrsg.), Modell- und computergestützte Unternehmensplanung, Verlag Gabler, Wiesbaden, 1973;

Haag, J.N., A macroscopic model of an interactive computing system, in: (KLERER-REINFELS,1968), 1968, p.285-285;

Händler, W., Automatentheoretischer Ansatz zur Behandlung von Teilnehmerrechensystemen, in: (HÄNDLER,1968a), 1968. p.267-288;

Händler, W. (Hrsg.), Teilnehmerrechensysteme, Verlag Oldenbourgh, München,1968a;

Hamilton, W.F., Moses, M.A., A computer-based corporate planning system, Management Science 21(1974), p.148-159;

Hansen, H.R. (Hrsg.), Computergestützte Marketingplanung, Verlag Moderne Industrie, München, 1974;

Hartmanis, H., Hopcroft, J.E., An overview of the theory of computational complexity, Journal of the ACM 18(1971), p.444-475;

Heinen, E., Das Zielsystem der Unternehmung, Verlag Gabler, Wiesbaden, 1966;

Heistand, R.E., An executive system implemented as a finite-state automaton, Communications of the ACM 7(1964), p.669-677;

Hermes, H., Einführung in die mathematische Logik, Teubner Verlag, Stuttgart, 1972;

Höns, G., Dialogsteuerung, Bericht Nr. 90 der Gesellschaft für Mathematik und Datenverarbeitung, Bonn, 1974;

Hoffmann, H.J., Winckler, G., Kontextsensitive Generierung beim Programmieren durch Auswählen, in: Brauer, M. (Hrsg.), GI-3,Jahrestagung, Lecture Notes in Computer Science 1(1973), Springer, Berlin, 1973, p.156-167;

Hopcroft, J.E., Ullman, J.D., Formal Languages and their Relation to Automata, Addison Wesley, London, 1969;

Hopcroft, J.E., Complexity of computer computations, Proceedings IFIP Congress 1974, Stockholm, p.620-626;

Horman, A.M., A man machine synergistic approach to planning and creative problem solving, Part 1, International Journal of Man-Machine Studies 3(1971), p.167-184;

Hultgren, G.V., Ergonometrische Forderungen beim Bildschirmeinsatz, Computerwoche, Heft 47(1975), p.4; Auszug aus: Hultgren, G.V., Arbeitsplätze mit Datenterminals, Datasaab GmbH, 1975;

ISIS Software Report, herausgegeben vom Infratest Information Sevice, München, 1976;

Johnson, C.J., Principles of interactive systems, IBM Systems Journal 7(1968), p.147-173;

Kahle, E., Betriebswirtschaftliches Problemlösungsverhalten, Verlag Gabler, Wiesbaden,1973;

Kieser, H.P., Mensch-Maschine-Kommunikation in Organisationen, Dissertation, Mannheim, 1973;

Kieser, A., Kubicek, H., Organisationsstruktur von Management-Informationssystemen, Zeitschrift für Betriebswirtschaft 44(1974), p.449-479;

Kirsch, W., Entscheidungsprozesse, Band I: Verhaltenswissenschaftliche Ansätze der Entscheidungstheorie, Verlag Gabler, Wiesbaden, 1970;

Kirsch, W., Entscheidungsprozesse, Band II: Informationsverarbeitungstheorie des Entscheidungsverhaltens, Verlag Gabler, Wiesbaden, 1971;

Kirsch, W., Heuristische Entscheidungsmodelle im Lichte der entscheidungs- und systemorientierten Betriebswirtschaftslehre, Einführung zu (KLEIN,1971), 1971a;

Klein, H., Heuristische Entscheidungsmodelle, Verlag Gabler, Wiesbaden, 1971;

Klerer, M., Reinfels, J. (Hrsg.), Interactive Systems for Experimental Applied Mathematics, Academic Press, New York, 1968;

Klingelhöfer, L., Personaleinsatzplanung, Dissertation, Göttingen, 1974;

Koreimann, S., Methoden und Organisation von Management-Informations-Systemen, de Gruyter, Berlin, 1971;

Krolak, P., Felts, W., Marble, G., A man machine approach towards solving the Travelling-Salesman-Problem, Communications of the ACM 14(1971), p.327-334;

Kupka, I., A structured model for dialog languages, in: Langmaack, H., Paul, M. (Hrsg.), 1. Fachtagung über Programmiersprachen, Lecture Notes in Economics and Mathematical Systems 75(1972), Springer, Berlin, 1972, p.157-168;

Kupka, I., Wilsing, N., A formal framework for dialog languages, Bericht Nr.2 des Inst. für Informatik der Universität Hamburg, 1972;

Kupka, I., Wilsing, N., An approach to formal semantics of dialog languages, Bericht Nr.4 des Inst. für Informatik der Inversität Hamburg, 1973;

Kupka, I., Wilsing, N., Functions describing interactive programming, in: Günther, A., et al. (Hrsg.), International Computing Symposium 1973, North-Holland, Amsterdam, 1974, p.41-45;

Kupka, I., Oberquelle, H., Wilsing, N., Projekt einer Rahmendialogsprache RDS, Mitteilung Nr.9 des Inst. für Informatik der Universität Hamburg, 1974;

Kupka, I., Wilsing, N., Dialogsprachen, Teubner Verlag, Stuttgart, 1975;

LeBeux, P., Selection processes and their relation to automata, Proceedings ACM National Conference 1973, p.336-341;

Levitt, K.N., The application of program-proving techniques to the verification of synchronization processes, Proceedings AFIPS Fall Joint Computer Conference, 1972, p.33-47;

Licklider, J.C.R., Man-computer symbiosis, IRE Transactions on Human Factors in Electronics 1(1960), p.4-11;

Licklider, J.C.R., Man-computer communication, Annual Review of Information Science and Technology 3(1968), p.201-240;

Linden, T.A., A summary of progress toward proving program correctness, Proceedings AFIPS Fall Joint Computer Conference 41, 1972, p.201-211;

Little, J.D.C., Models and managers: the concept of a decision calculus, Management Science 16(1970), p.B466-B485;

Little, J.D.C., Ein On-Line Marketing-Mix Modell, in: (HANSEN,1974), 1974, p.651-704;

Lodish, L.M., CALLPLAN: An interactive salesman's call planning system, Management Science 13(1971), p.P25-P40;

Losbichler, B., Mühlbacher, J., Entwicklung eines Mensch-Maschine-Kommunikationssystems durch Top-Down-Entwurf, Angewandte Informatik 16(1974), p.453-463;

Manna, Z., The correctness of programs, Journal of Computer and System Sciences 3(1969), p.119-127;

Manna, Z., The correctness of nondeterministic programs, Artificial Intelligence 1(1970), p.1-26;

Manna, Z., Mathematical theory of partial correctness, Journal of Computer and System Sciences 5(1971). p.239-253;

Martin, J., Design of Man-Computer Dialogues, Prentice Hall, Englewood Cliffs, 1973;

Meadow, C.T., Man-Machine Communication, John Wiley, New York, 1970;

Mertens, P., Kress, H., Mensch-Maschinen-Kommunikation als Hilfe bei Entscheidungsvorbereitung und Planung, Zeitschrift für betriebswirtschaftliche Forschung 22 (1970), p.1-21;

Mertens, P., Industrielle Datenverarbeitung, Band I, Verlag Gabler, Wiesbaden, 1972;

Mertens, P., Griese, J., Industrielle Datenverarbeitung, Band II, Verlag Gabler, Wiesbaden, 1972;

Mevert, P., Dickson, G.W., Short-term planning: an interactive modelling approach, in: (GROCHLA-SZYPERSKI,1973), 1973, p.621-635;

Miller, R.B., Archtypes in man-computer problem solving, IEEE Transactions on Man-Machine Studies 10(1969), p.219-241;

Morton, M.S., McCosh, A.M., Terminal costing for better decisions, Harvard Business Review 46(1968), p.147-156;

Morton, M.S., Management-Entscheidungen im Bildschirmdialog, Verlag Girardet, Essen, 1972;

Mühlbacher, J., Mensch-Maschinen-Kommunikation bei der Segmentierung von Programmen, Angewandte Informatik 14(1972), p.389-394;

Mumford, E., Sackman, H. (Hrsg.), Human Choice and Computers, Proceedings IFIP Conference on Human Choice and Computers, Wien 1974, Nort-Holland, Amsterdam, 1975;

Naur, P., Proof of algorithms by general snapshots, BIT (nordisk Tidskrift for Informations Behandling) 6(1966), p.310-316;

Newell, A., Simon, H.A., Human Problem Solving, Prentice Hall, Englewood Cliffs, 1972;

Newman, W.M., A system for interactive graphical programming, Proceedings AFIPS Spring Joint Computer Conference, 1968, p.47-54;

Nickerson, R.S., Man-computer interaction: a challenge for human factors research, IEEE Transactions on Man-Machine Systems 10(1969), p.164-180;

Nilsson, N.J., Problem-Solving Methods in Artificial Intelligence, McGraw-Hill, New York, 1971;

Noltemeier, H., Graphentheorie mit Algorithmen und Anwendungen, de Gruyter, Berlin, 1976;

Noltemeier, H. (Hrsg.), Computergestützte Planungssysteme, Physica, Würzburg, 1976a;

Oberquelle, H., Dialogsysteme aus der Sicht des Benutzers - Grundlagen für eine transparente Beschreibung und Begriffsbildung, ACM German Chapter Lectures II, 1975, p.25-38;

Parnas, D.L., On the use of transition diagrams in the design of a user interface for an interactive computer system, Proceedings ACM National Conference 24(1969), p.379-385;

Petri, C.A., Kommunikation mit Automaten, Dissertation, Bonn, 1962;

Poensgen, O.H., Wirtschaftliche Vor- und Nachteile von Teinehmersystemen, in: (GROCHLA,1970a), 1970, p.97-120;

Press, L., Toward balanced man-machine systems, International Journal of Man-Machine Studies 3(1971), p.61-73;

Rechenberg, P., Was ist Backtrack-Programmierung?, Elektronische Rechenanlagen 17(1975), p.276-282;

Ribbeck, W., Grundlagen der Time-Sharing-Anwendung, VDI-Verlag, Düsseldorf, 1973;

Rogers, H., Theory of Recursive Functions and Effective Computability, McGraw-Hill, New York, 1967;

Roth, H., Ein Modell zur Beschreibung interaktiver Problemlösungen, ACM German Chapter Lectures II, 1975a , p.39-50;

Roth, H., Ein auf Turingmaschinen basierendes Modell zur Beschreibung interaktiver Problemlösungen, Disc. Pap. 7503 des Lehrst. f. math. Verfahrensforschung und Datenverarbeitung der Universität Göttingen, 1975b;

Roth, H., Semantische Aspekte der Programmoptimierung, in: Siefkes, D. (Hrsg.), GI-4.Jahrestagung, Lecture Notes in Computer Science 26(1975), Springer, Berlin, 1975c;

Roy, B., Problems and methods with multiple objective functions, Mathematical Programming 1(1971), p.239-266;

Sackman, H., Man-Computer Problem Solving, Auerbach, Princeton, 1970;

Schmid, H.A., Zur Struktur von Dialogsystemen, Dissertation, Karlsruhe, Interner Bericht 74/8 des Instituts für Praktische Mathematik der Universität Karlsruhe, 1974;

Schober, F., Eine interaktive ökonometrische Modellsprache in APL, in: (HANSEN,1974), 1974, p.417-445;

Schur, L.D., Time-Shared Computer Languages: An Introduction to Conversational Programming, Addison Wesley, Reading Mass.,1973;

Sherwood, F.D. (Hrsg.), Interactive Computing, Infotech. Information, Maidenhead, 1972;

Simmonard, M., Linear Programming, Prentice Hall, Englewood Cliffs, 1966;

Simon, H.A., Perspektiven der Automation für Entscheider, Verlag Schnelle, Quickborn, 1966;

Simon, H.A., The theory of problemsolving, Proceedings IFIP Congress 1971, Ljubljana, p.261-277;

Szyperski, N., Vorgehensweise bei der Gestaltung computergestützter Entscheidungssysteme, in: (GROCHLA,1971a), 1971, p.37-64;

Szyperski, N., Winand, U., Wirtschaftliche Aspekte des Einsatzes von Time-Sharing-Systemen in Problemlösungsprozessen, Data Scope 9(1972), p.3-13;

UNIVAC, NUALGOL Programmer Reference, UNIVAC 1100 Series, Sperry Rand, 1971;

UNIVAC, BASIC, Programmer Reference, UNIVAC 1100 Series, UP-7925, Sperry Rand, 1972;

UNIVAC, APL/1100 System Summary, Sperry Rand, 1973;

Wacker, W.H., Betriebswirtschaftliche Informationstheorie, Westdeutscher Verlag, Opladen, 1971;

Wedekind, H., Datenbanksysteme, Band I, Bibliographisches Institut, Mannheim, 1975;

Wegner, P., Operational semantics of programming languages, Proceedings of an ACM Conference on Proving Assertions about Programs, SIGPLAN Notices 7,1(1972), p.128-141;

Wild, J., Zur Problematik der Nutzenbewertung von Informationen, Zeitschrift für Betriebswirtschaft 41(1971), p.315-335;

Winand, U., Konzeption modellgestützter Problemlösungen, Angewandte Informatik 17(1975), p.206-212;

Wirth, N., Systematisches Programmieren, Teubner Verlag, Stuttgart, 1972

Witte, E., Das Informationsverhalten in Entscheidungsprozessen, Verlag Mohr, Stuttgart, 1972;

Wöhe, G., Einführung in die allgemeine Betriebswirtschaftslehre, Verlag Vahlen, München, 1973/74;

Zahn, E., Zur Anwendung mathematischer Modelle bei der strategischen Planung, in: (GROCHLA-SZYPERSKI,1973), 1973, p.597-619;

Zangemeister, C., Werturteil und formalisierte Planungsprozesse, in: (GROCHLA-SZYPERSKI,1973), 1973, p.97-123.

(Abk.: ACM = Association for Computing Machinery,
AFIPS = American Federation of Information Processing Societies,
IEEE = Institute of Electrical and Electronic Engineers,
IFIP = International Association for Information Processing.)

Notation

Für die wichtigsten verwendeten Symbole gibt die folgende Liste an, auf welcher Seite sie eingeführt werden:

A, A^*, A^+	59
AP	101
$B_Q, (q)$	72
$\bigvee$-BRANCH	102
c	72
d	59
$diag_T$	84
D_x , D_y , D_z	101
DEF(..)	95
G(T)	71
HALT	102
$\mathbb{N}$	60
P^+	59
P_T	73
q_o	59
(q,w,a(w))	60
Q, Q_U	59
Q_F	90
SELECT	102
START	101
t	61
T	59
T^F	96
W(q,q')	72
W_t, W_T	84

Anhang 1 Die implementierte Transportmitteldisposition - Problemlösung

Die implementierte Problemlösung der Transportmitteldisposition aus 2.1.1., auf die sich in 3.1.2., 3.2.2. und 3.4.3. bezogen wird, wurde auf einem Rechner UNIVAC 1108[1] unter dem Betriebssystem EXEC 8 zur Benutzung von einem Bildschirmgerät[2] aus implementiert. Sie besteht aus dem nachfolgend aufgelisteten NUALGOL-Programm TRANSPORT.VERGABE, das - neben NUALGOL-Anweisungen[3] - drei externe Prozeduren (ERTREA,ERCSFD und TPL) und die Datei DATENBANK benutzt.

ERTREA(Frage,Antwort,label) - Frage und Antwort STRING-Variable, label eine Marke - ist eine Fortran-Prozedur, die den Inhalt des Feldes Frage auf dem Bildschirm ausgibt, die Textanfangsmarke direkt dahinter stehen läßt und einen danach eingegebenen Text in das Feld Antwort einliest; bei Eingabe von ℘EOF wird zur Marke label gesprungen.

ERCSFD(Steueranweisung) ist eine FORTRAN-Prozedur, die das Absetzen von bestimmten Anweisungen aus der Steuersprache des Betriebssystems in einem in TRANSPORT.VERGABE geführten Lösungsprozeß ermöglicht, die dann sofort ausgeführt werden; hier wird damit die Datei DATENBANK mit den Daten

---

[1] Rechenanlage UNIVAC 1108 der Gesellschaft für wissenschaftliche Datenverarbeitung mbH Göttingen.

[2] Zur Zeit der Erstellung UNISCOPE 100/200.

[3] Vgl. (UNIVAC,1971).

über Transportmittel und Kundenbestellungen[1] für einen Lösungsprozeß zugeordnet. TPL ist eine NUALGOL-Prozedur, die im wesentlichen aus einem Transportalgorithmus und der Ausgabe einer erstellten optimalen Lösung besteht; TPL wird im Anschluß an die Liste der Anweisungen des Programmes TRANSPORT.VERGABE kurz skizziert.

Die Problemlösungslogik des Programmes TRANSPORT.VERGABE wird aus Figur 2.1.1.1. deutlich; Dialogaktivitäten des Rechners[2] - durch Rhombus-Umrahmung gekennzeichnet - sind dort wörtlich wie im Programm (großgeschrieben) oder sinngemäß verkürzt (Normalschrift) wiedergegeben; Benutzeraktivitäten sind dort - bis auf START - stets einfach als "Antworteingabe"[3] bezeichnet. Da das Programm mit Figur 2.1.1.1. und den vorangegangenen Erläuterungen zu den verwendeten externen Prozeduren relativ leicht verständlich ist, soll hier keine ausführliche Dokumentation seiner einzelnen Komponenten, Variable, etc., vorgenommen werden; einige Anmerkungen erscheinen jedoch noch angebracht:

---

[1] Vgl. 2.1.1.

[2] Vgl. "$diag_T$" in 3.1.1.

[3] Darin kommt der etwas zu einem "Sklavensystem" (vgl. MERTENS-KRESS,1970), p.2) hin tendierende Charakter der implementierten Problemlösung zum Ausdruck.

1. Im wesentlichen aus Gründen der Übersichtichkeit und des Beispielcharakters des Programmes sind die folgenden, für ein Anwendungsprogramm wichtigen Gesichtspunkte weitgehend vernachlässigt worden:
   a. Implementierung einer "Hilfe-Funktion"[1];
   b. Möglichkeiten zum Verkürzen von Lösungsprozessen (für geübte Benutzer);
   c. ausführliche Fehlerdiagnose mit Hilfeangeboten bei fehlerhaften Benutzeraktivitäten[2];
   d. geeignete Verfügung über in einem Lösungsprozeß erarbeitete Daten[3].
2. Sind die Datei DATENBANK verfügbar und das Programm TRANSPORT.VERGABE einschließlich der verwendeten externen Prozeduren gestartet worden, so erfordert die Durchführung eines Lösungsprozesses - vom Rechner her gesehen - nur Bedienungskenntnis des benutzten Bildschirmgerätes.
3. Die Hardware und das Betriebssystem EXEC 8 des Rechners UNIVAC 1108 erlauben starke Unterbrechungen[4] an nahezu jeder Stelle eines Lösungsprozesses.

Die Liste der Anweisungen des Programmes TRANSPORT.VERGABE:

---

[1] Vgl. (KUPKA-WILSING,1975), p.42.

[2] Vgl. dazu (MARTIN,1973), p.449ff.

[3] Vgl Zeilen 117/118 in der folgenden Programmliste.

[4] Vgl. 2.2.(iii)2.

```
TRANSPORT.VERGABE

 1 BEGIN
 2 INTEGER M,N,I;
 3 EXTERNAL FORTRAN PROCEDURE ERTREA, CROSFD;
 4 EXTERNAL PROCEDURE TPL;
 5 CROSFD('BASIC.A DATENBANK . ');
 6 READ(FILE('DATENBANK'),M,N,I,Q1);
 7 G1: BEGIN
 8 INTEGER ARRAY X(1:M+1,0:N+1),B(1:M),F(1:M),K(1:N),E(0:N),KNEU(0:N);
 9 STRING FRAGE(60), ANTWORT1(A1(1),REST(9)), ANTWORT2(A2(1),REST2(9));
10 INTEGER J,HV1,BSUMME,KSUMME,MS,NS;
11 FORMAT F1(S1,I:N:(S2,I3),A), F2(I:N:(I5),A), F3(S1,I:M:(S2,I3),A),
12        F4(I:M:(I5),A),F5(A,S1), F6(S27,I5,A), F7(S1,I5,S45,A);
13 MS = M + 1;
14 NS = N + 1;
15 READ(FILE('DATENBANK'),B,P,K,E,Q2);
16 Q2: FOR I = (1,1,M) DO X(I,NS) = B(I);
17 FOR J = (1,1,N) DO KNEU(J) = X(MS,J) = K(J);
18 FOR I = (1,1,M) DO BSUMME = BSUMME + B(I);
19 FOR J = (1,1,N) DO KSUMME = KSUMME + KNEU(J);
20 X(MS,0) = KSUMME; X(MS,NS) = BSUMME;
21 ZULAESSIGKEIT:
22 IF BSUMME LSS X(MS,0) THEN
23   BEGIN
24   TEILLIEFERUNGSENTSCHEIDUNGEN:
25   WRITE('DIE KAPAZITAETEN DER ANGEBOTENEN TRANSPORTMITTEL SIND (NOCH)');
26   WRITE('UNZUREICHEND. WOLLEN SIE                                    ');
27   WRITE('TEILLIEFERUNGEN          - EINGABE EINER 1  -               ');
28   WRITE('BESTIMMEN ODER DEN DISPOSITIONSPROZESS VORERST              ');
29   WRITE('ABBRECHEN                - EINGABE EINER 2  ?               ');
30   READ(F5,A1,FEHLER1,FEHLER1,FEHLER1);
31   IF A1 EQL '1' THEN
32     BEGIN
33     WRITE('FOLGENDE (BEREITS REDUZIERTE) BESTELLUNGEN LIEGEN VOR:');
34     WRITE(F1,' ',FOR J = (1,1,N) DO (' K',J));
35     WRITE(F2,FOR J = (1,1,N) DO (KNEU(J)));
36     HV1 = X(MS,0) - BSUMME;
37     WRITE(F7,'(',HV1,' ME KOENNEN (NOCH) NICHT AUSGELIEFERT WERDEN)');
38     FRAGE = 'WELCHER KUNDE SOLL NICHT VOLL BELIEFERT WERDEN (KDNR.)?';
39     ERTREA(FRAGE,ANTWORT2,EXIT);
40     J = INT(ANTWORT2);
41     IF J LEQ 0 OR J GTR N THEN GOTO FEHLER1;
42     WRITE(F5,'WIEVIEL SOLL AN DEN KUNDEN ',J);
43     FRAGE = 'GELIEFERT WERDEN?';
44     ERTREA(FRAGE,ANTWORT2,EXIT);
45     HV1 = INT(ANTWORT2);
46     IF HV1 LSS 0 OR HV1 GTR 99999 OR HV1 GTR KNEU(J) THEN GOTO FEHLER1;
47     X(MS,0) = X(MS,0) - KNEU(J);
48     X(MS,J) = KNEU(J) = HV1;
49     X(MS,0) = X(MS,0) + KNEU(J);
50     GOTO ZULAESSIGKEIT;
51     END
52   ELSE IF A1 EQL '2' THEN
53     BEGIN
54     WRITE('DER DISPOSITIONSPROZESS WIRD VORERST ABGEBROCHEN.');
55     GOTO EXIT;
56     END
57     ELSE
58     FEHLER1:
59     BEGIN
60     WRITE('LETZTE ANGABE FEHLERHAFT, DAHER NICHT BERUECKSICHTIGT');
```

```
 61      GOTO TEILLIEFERUNGSENTSCHEIDUNGEN;
 62      END
 63    END;
 64  FRAGEN1:
 65  FRAGE = 'WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN?';
 66  ERTREA(FRAGE,ANTWORT1,EXIT);
 67  IF A1 EQL 'J' THEN
 68    TRANSPORTMITTELVORGABE:
 69    BEGIN
 70    WRITE('MOEGLICHE TRANSPORTMITTEL, (REC.)KAPAZITAETEN, KOSTENSAETZE:
 71    WRITE(F3,' ',FOR I = (1,1,M) DO (' T',I));
 72    WRITE(F4, FOR I = (1,1,M) DO (X(I,NS)));
 73    WRITE(F4, FOR I = (1,1,M) DO (P(I)));
 74    WRITE('KUNDEN, (REC.)BESTELLMENGEN, ENTFERNUNGEN:');
 75    WRITE(F1,' ', FOR J = (1,1,N) DO (' K',J));
 76    WRITE(F2, FOR J = (1,1,N) DO (X(MS,J)));
 77    WRITE(F2, FOR J = (1,1,N) DO (E(J)));
 78    WRITE('VERGEBEN SIE (WEITERE) TRANSPORTMITTEL DURCH ANGABEN UNTER:'
 79    WRITE('TRANSP.MITTELNR.   KUNDENNR.   ZU TRANSP. MENGE       ');
 80    READ(I,J,HV1,FEHLER2,FEHLER2,FEHLER2);
 81    IF I GTR M OR J GTR N OR HV1 GTR MIN(X(I,NS),X(MS,J))
 82    OR I LEQ 0 OR J LEQ 0 OR HV1 LSS 0 THEN
 83    FEHLER2:
 84      BEGIN
 85      WRITE('ANGABEN FEHLERHAFT ODER UNMOEGLICH, NICHT BERUECKSICHTIGT.
 86      GOTO TRANSPORTMITTELVORGABE;
 87      END;
 88    X(I,J) = HV1;
 89    X(I,NS) = X(I,NS) - HV1;
 90    X(MS,J) = X(MS,J) - HV1;
 91    X(MS,0) = X(MS,0) - HV1;
 92    X(MS,NS) = X(MS,NS) - HV1;
 93
 94    FRAGEN:
 95    FRAGE = 'WEITERE TRANSPORTMITTELVERGABE GEWUENSCHT?';
 96    ERTREA(FRAGE,ANTWORT2,EXIT);
 97    IF A2 EQL 'J' THEN GOTO TRANSPORTMITTELVORGABE;
 98    IF A2 NEQ 'N' THEN
 99      BEGIN
100      WRITE('LETZTE ANTWORT FEHLERHAFT, DAHER NICHT BERUECKSICHTIGT.');
101      GOTO FRAGEN;
102      END
103    ELSE GOTO TRANSPORTVERGABE;
104
105    END;
106  IF A1 EQL 'N' THEN
107    TRANSPORTVERGABE:
108    BEGIN
109    X(MS,0) = X(MS,NS) - X(MS,0);
110    TDL(M,N,P,E,X);
111    FRAGE = 'AKZEPTIEREN SIE DIESE TRANSPORTMITTELVERGABE?';
112    ERTREA(FRAGE,ANTWORT2,EXIT);
113    IF A2 EQL 'J' THEN
114      BEGIN
115      WRITE('TRANSPORTMITTELDISPOSITION BEENDET.');
116      COMMENT ************************************************
117              * HIER KOENNEN NUN ERSTELLUNG VON VERSANDDOKUMENTEN,  *
118              * U.AE., VGL. (MERTENS,1972), P.75FF., ERFOLGEN.      *
119              ************************************************;
120      END
121    ELSE
122      BEGIN
```

```
123     WRITE('DIE LETZTE TRANSPORTMITTELVERGABE WIRD VERWORFEN.');
124     WRITE('SIE KOENNEN NOCHMALS TRANSPORTMITTEL VORGEBEN.');
125     FOR I := (1,1,MS) DO FOR J := (0,1,NS) DO X[I,J] := 0;
126     FOR I := (1,1,M) DO X[I,NS] := B[I];
127     FOR J := (1,1,N) DO
128       BEGIN
129       X[MS,J] := KNEU[J];
130       X[MS,0] := X[MS,0] + KNEU[J];
131       END;
132     X[MS,NS] := BSUMME;
133     GOTO FRAGEN1;
134     END
135   END
136 ELSE
137   BEGIN
138   WRITE('LETZTE ANTWORT FEHLERHAFT,DAHER NICHT BERUECKSICHTIGT.');
139   GOTO FRAGEN1;
140 END;
141 EXIT:
142 END;
143 END;
```

Die externe NUALGOL-Prozedur TPL(M,N,P,E,X) hat im Überblick die folgenden Funktionen (M Anzahl der Transportmittel, N Anzahl der Kunden, P Kostensatzvektor der Transportmittel, E Entfernungsvektor der Kunden, X eine (M+1, N+2)-Matrix, die insbesondere in den Elementen x_{ij}, $1 \leq i \leq M$, $1 \leq j \leq N$, von einem Benutzer vorgegebene Transporte mit Transportmittel T_i zum Kunden K_j enthält) :

1. Erstellung eines Transportproblems, in dem nur Transportmittel mit positiver Kapazität und Kunden mit positiver Bestellmenge vorhanden sind; insbesondere durch die Möglichkeit zur Vergabe von Transporten vor der Berechnung einer optimalen Lösung kann der Fall

auftreten, daß die Kapazität von Transportmitteln bereits erschöpft ist oder die volle Belieferung von Kunden schon feststeht.

2. Bestimmung einer optimalen Lösung mit dem "Stepping-Stone-Algorithmus"[1]; eine zulässige Ausgangsbasislösung wird dabei nach dem "Gesamtminimumverfahren"[2] erstellt.

3. Ausgabe einer gefundenen optimalen Basislösung zusammen mit den von einem Benutzer zuvor festgelegten Transporten in der aus dem Dialogprotokoll aus Anhang 2 ersichtlichen Form.

Eine detaillierte Dokumentation der externen Prozedur TPL erscheint im Rahmen dieser Arbeit nicht notwendig.

Die Liste der Anweisungen von TPL(M,N,P,E,X) :

---

[1] Vgl. (SIMMONARD,1966), p.239/240; auf eine besondere Behandlung von Degeneration wurde verzichtet, da - (SIMMONARD,1966), p.243 - "No practical example of a transportation problem with cycling is known".

[2] Vgl. (SIMMONARD,1966), p.240/241, dort "method of minimal cost" genannt.

```
PROCEDURE TPL(M,N,P,E,X);
INTEGER M,N;
INTEGER ARRAY P,E,X;
BEGIN
INTEGER MA,NA,MS,NS,I,J;
INTEGER ARRAY MERKZEI(1:M), MERKSPA(0:N);

COMMENT  ************************************************************
         *   ERSTELLUNG EINES TRANSPORTPROBLEMS MIT POSITIVEN       *
         *   ANGEBOTEN UND NACHFRAGEN                               *
         ************************************************************;

FOR I = (1,1,M) DO IF X(I,N+1) NEQ 0 THEN MERKZEI(I) = MA = MA + 1;
FOR J = (0,1,N) DO IF X(M+1,J) NEQ 0 THEN MERKSPA(J) = NA = NA + 1;
MS = MA + 1;
NS = NA + 1;
BEGIN
INTEGER ARRAY Z(1:MS,1:NS),C(1:MA,1:NA),CQUER(1:MA,1:NA),
              U(1:MS),V(1:NS),G(1:MS,1:NS);
INTEGER HV1,HV2,S,T,AENDERUNG;
BOOLEAN ARRAY ZEILE(1:MS),ZEILER(1:MS),SPALTE(1:NS),SPALTER(1:NS);
BOOLEAN SCH1,SCH2;
FOR I = (1,1,M) DO
  BEGIN
  HV1 = MERKZEI(I);
  FOR J = (0,1,N) DO
    BEGIN
    HV2 = MERKSPA(J);
    IF HV1 NEQ 0 AND HV2 NEQ 0 THEN
      BEGIN
      C(HV1,HV2) = CQUER(HV1,HV2) = P(I) * E(J);
      Z(HV1,HV2) = -1;
      END;
    END;
  END;
FOR I = (1,1,M) DO IF MERKZEI(I) NEQ 0 THEN Z(MERKZEI(I),NS) = X(I,N+1);
FOR J = (0,1,N) DO IF MERKSPA(J) NEQ 0 THEN Z(MS,MERKSPA(J)) = X(M+1,J);

COMMENT  ************************************************************
         *    BERECHNUNG EINER ZULAESSIGEN AUSGANGSLOESUNG        *
         ************************************************************;

AUSGANGSLOESUNG:
FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
IF ZEILE(I) OR SPALTE(J) THEN ELSE
  BEGIN
  IF NOT SCH1 THEN
    BEGIN
    HV1 = CQUER(I,J);
    SCH1 = TRUE;
    END;
  IF CQUER(I,J) LEQ HV1 THEN
    BEGIN
    HV1 = CQUER(I,J);
    S = I;
    T = J;
    END;
  END;
IF SCH1 THEN
  BEGIN
```

```
 61   IF NOT SCH2 THEN
 62     BEGIN
 63     Z(S,T) = MIN(Z(S,NS),Z(MS,T));
 64     IF Z(S,NS) EQL Z(MS,T) THEN SCH2 = ZEILE(S) = TRUE;
 65     END
 66   ELSE
 67     BEGIN
 68     Z(S,T) = 0;
 69     SCH2 = FALSE;
 70     END;
 71   IF Z(S,T) EQL Z(S,NS) THEN ZEILE(S) = TRUE
 72   ELSE SPALTE(T) = TRUE;
 73   QUER(S,T) = 99999;
 74   G(S,T) = 1;
 75   Z(S,NS) = Z(S,NS) - Z(S,T);
 76   Z(MS,T) = Z(MS,T) - Z(S,T);
 77   SCH1= FALSE;
 78   ZEILE(MS) = SPALTE(NS) = TRUE;
 79   FOR I = (1,1,MA) DO ZEILE(MS) = ZEILE(MS) AND ZEILE(I);
 80   FOR J = (1,1,NA) DO SPALTE(NS) = SPALTE(NS) AND SPALTE(J);
 81   IF NOT ZEILE(MS) OR NOT SPALTE(NS) THEN GOTO AUSGANGSLOESUNG;
 82   END;
 83
 84 COMMENT  ************************************************************
 85          *   U(I)-V(J)-BESTIMMUNG                                   *
 86          ************************************************************
 87
 88 FOR J = (1,1,NS) DO SPALTE(J) = FALSE;
 89 V(NA) = 0;
 90 SPALTE(NA) = TRUE;
 91 BESTIMMUNG:
 92 FOR I = (1,1,MA) DO ZEILE(I) = FALSE;
 93 FOR J = (1,1,NA) DO IF SPALTE(J) THEN FOR I = (1,1,MA) DO
 94   BEGIN
 95   IF (Z(I,J) NEQ -1) AND NOT ZEILER(I) THEN
 96     BEGIN
 97     U(I) = C(I,J) - V(J);
 98     ZEILE(I) = ZEILER(I) = TRUE;
 99     END;
100   END;
101 ZEILER(MS) = SPALTER(NS) = TRUE;
102 FOR I = (1,1,MA) DO ZEILER(MS) = ZEILER(MS) AND ZEILER(I);
103 FOR J = (1,1,NA) DO SPALTER(NS) = SPALTER(NS) AND SPALTER(J);
104 IF ZEILER(MS) AND SPALTER(NS) THEN GOTO OPTPRUEFUNG;
105 FOR J = (1,1,NA) DO SPALTE(J) = FALSE;
106 FOR I = (1,1,MA) DO IF ZEILE(I) THEN FOR J = (1,1,NA) DO
107   BEGIN
108   IF (Z(I,J) NEQ -1) AND NOT SPALTER(J) THEN
109     BEGIN
110     V(J) = C(I,J) - U(I);
111     SPALTE(J) = SPALTER(J) = TRUE;
112     END;
113   END;
114 ZEILER(MS) = SPALTER(NS) = TRUE;
115 FOR I = (1,1,MA) DO ZEILER(MS) = ZEILER(MS) AND ZEILER(I);
116 FOR J = (1,1,NA) DO SPALTER(NS) = SPALTER(NS) AND SPALTER(J);
117 IF ZEILER(MS) AND SPALTER(NS) THEN ELSE GOTO BESTIMMUNG;
118
119
120 COMMENT  ************************************************************
121          *   PRUEFUNG AUF OPTIMALITAET                              *
122          ************************************************************
```

```
123
124 OPTIMUEFUNG:
125 SCH1 = SCH2 = TRUE;
126 FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
127   BEGIN
128   IF SCH2 THEN
129     BEGIN
130     HV1 = U(I) + V(J) - C(I,J);
131     S = I;
132     T = J;
133     SCH2 = FALSE;
134     END
135   ELSE
136     BEGIN
137     HV2 = U(I) + V(J) - C(I,J);
138     IF HV2 GTR 0 THEN SCH1 = FALSE;
139     IF HV2 GTR HV1 THEN
140       BEGIN
141       HV1 = HV2;
142       S = I;
143       T = J;
144       END;
145     END
146   END;
147 IF SCH1 THEN GOTO OPTIMALESX;
148
149 COMMENT  ************************************************************
150          * BESTIMMUNG DES GESCHLOSSENEN WEGES ZUR VERAENDERUNG      *
151          * DER AKTUELLEN BASISLOESUNG                               *
152          ************************************************************;
153
154 G(S,T) = 1;
155 FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
156   BEGIN
157   G(I,NS) = G(I,NS) + G(I,J);
158   G(MS,J) = G(MS,J) + G(I,J);
159   END;
160 RANGVERAENDERUNG:
161 FOR I = (1,1,MA) DO IF G(I,NS) EQL 1 THEN
162 FOR J = (1,1,NA) DO IF G(I,J) EQL 1 THEN
163   BEGIN
164   G(I,NS) = 0;
165   G(MS,J) = G(MS,J) - 1;
166   G(I,J)  = 0;
167   END;
168 FOR J = (1,1,NA) DO IF G(MS,J) EQL 1 THEN
169 FOR I = (1,1,MA) DO IF G(I,J) EQL 1 THEN
170   BEGIN
171   G(MS,J) = 0;
172   G(I,NS) = G(I,NS) - 1;
173   G(I,J)  = 0;
174   END;
175 FOR I = (1,1,MA) DO IF G(I,NS) EQL 1 THEN GOTO RANGVERAENDERUNG;
176 FOR J = (1,1,NA) DO IF G(MS,J) EQL 1 THEN GOTO RANGVERAENDERUNG;
177 HV1 = S;
178 HV2 = T;
179 G(S,NS) = G(S,NS) - 1;
180 G(MS,T) = G(MS,T) - 1;
181 ZYKELMARKIERUNG:
182 FOR I = (1,1,MA) DO IF G(I,T) EQL 1 AND I NEQ S THEN
183   BEGIN
184   G(I,T) = -1;
```

```
185   S = I;
186   G(MS,T) = G(MS,T) - 1;
187   G(S,NS) = G(S,NS) - 1;
188   GOTO ZYKEL1;
189   END;
190 ZYKEL1:
191 FOR J = (1,1,NA) DO IF G(S,J) EQL 1 AND J NEQ T THEN
192   BEGIN
193   T = J;
194   G(S,NS) = G(S,NS) - 1;
195   G(MS,T) = G(MS,T) - 1;
196   GOTO ZYKEL2;
197   END;
198 ZYKEL2:
199 FOR I = (1,1,MA) DO IF G(I,NS) GTR 0 THEN GOTO ZYKELMARKIERUNG;
200 FOR J = (1,1,NA) DO IF G(MS,J) GTR 0 THEN GOTO ZYKELMARKIERUNG;
201
202 COMMENT  ************************************************************
203           * VERAENDERUNG DER AKTUELLEN BASISLOESUNG >ENTLANG> DES *
204           * BESTIMMTEN GESCHLOSSENEN WEGES                        *
205           ************************************************************;
206
207 IF HV1 NEQ S OR HV2 NEQ T THEN GOTO ZYKELMARKIERUNG;
208 AENDERUNG = 999999;
209 FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
210 IF G(I,J) EQL -1 AND (I NEQ S OR J NEQ T) THEN
211   BEGIN
212   IF Z(I,J) LSS AENDERUNG THEN
213     BEGIN
214     HV1 = I;
215     HV2 = J;
216     AENDERUNG = Z(I,J);
217     END;
218   END;
219 Z(S,T) = 0;
220 Z(HV1,HV2) = -1;
221 G(HV1,HV2) = 0;
222 FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
223   BEGIN
224   IF G(I,J) EQL 1 THEN Z(I,J) = Z(I,J) + AENDERUNG;
225   IF G(I,J) EQL -1 THEN Z(I,J) = Z(I,J) - AENDERUNG;
226   IF Z(I,J) EQL -1 THEN G(I,J) = 0 ELSE G(I,J) = 1;
227   END;
228 FOR I = (1,1,MS) DO
229   BEGIN
230   G(I,NS) = U(I) = 0;
231   ZEILER(I) = FALSE;
232   END;
233 FOR J = (1,1,NS) DO
234   BEGIN
235   G(MS,J) = V(J) = 0;
236   SPALTER(J) = SPALTE(J) =  FALSE;
237   END;
238 V(NA) = 0;
239 SPALTE(NA) = TRUE;
240 GOTO BESTIMMUNG;
241
242 COMMENT  ************************************************************
243           *   AUSGABE EINER OPTIMALEN LOESUNG                       *
244           ************************************************************;
245
246 OPTIMALESX:
```

```
247 FOR I = (1,1,MA) DO FOR J = (1,1,NA) DO
248   BEGIN
249   FOR HV1 = (1,1,M) DO IF MERKZEI(HV1) EQL I THEN S = HV1;
250   FOR HV2 = (0,1,N) DO IF MERKSPA(HV2) EQL J THEN T = HV2;
251   IF Z(I,J) NEQ -1 THEN X(S,T) = X(S,T) + Z(I,J);
252   END;
253 WRITE('EINE KOSTENOPTIMALE TRANSPORTMITTELDISPOSITION - RELATIV ZU');
254 WRITE('VON IHNEN GETROFFENEN TRANSPORTMITTELVORGABEN - IST:');
255   BEGIN
256   INTEGER ARRAY AUSGABE(0:M+1,0:N+1);
257   FORMAT F1(S25,I4,S10,A),F2(S1,:HV1:(S2,I3),A),F3(:HV1:(I5),A);
258   FOR I = (1,1,M) DO
259   BEGIN
260   SCH1 = TRUE;
261   WRITE(F1,'                              T',I,' GELIEFERT');
262   FOR J = (1,1,N) DO AUSGABE(0,J) = 0;
263   HV1 = 0;
264   FOR J = (1,1,N) DO IF X(I,J) NEQ 0 THEN
265     BEGIN
266     SCH1 = FALSE;
267     HV1 = HV1 + 1;
268     AUSGABE(0,HV1) = J;
269     AUSGABE(I,HV1) = X(I,J);
270     END;
271   IF NOT SCH1 THEN
272     BEGIN
273     WRITE(F2,' ',FOR J = (1,1,HV1) DO (' K',AUSGABE(0,J)));
274     WRITE('MIT (IN ME):');
275     WRITE(F3,FOR J = (1,1,HV1) DO (AUSGABE(I,J)));
276     END
277   ELSE WRITE('KEINEN KUNDEN');
278   END;
279   END;
280 END;
281 END;
282 END;
```

Anhang 2 Dialogprotokoll eines Transportmitteldisposition-Lösungsprozesses

```
DIE KAPAZITAETEN DER ANGEBOTENEN TRANSPORTMITTEL SIND (NOCH)
UNZUREICHEND. WOLLEN SIE
TEILLIEFERUNGEN          - EINGABE EINER 1  -
BESTIMMEN ODER DEN DISPOSITIONSPROZESS VORZEITIG
ABBRECHEN                - EINGABE EINER 2  ?
1
FOLGENDE (BEREITS REDUZIERTE) BESTELLUNGEN LIEGEN VOR:
  K  1 K  2 K  3 K  4 K  5 K  6
  200  150  100  300  250  400
(  100 ME KOENNEN (NOCH) NICHT AUSGELIEFERT WERDEN)
WELCHER KUNDE SOLL NICHT VOLL BELIEFERT WERDEN (KDNR.)?  3
WIEVIEL SOLL AN DEN KUNDEN     3
GELIEFERT WERDEN?     0
WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN?  H
LETZTE ANTWORT FEHLERHAFT,DAHER NICHT BERUECKSICHTIGT.
WOLLEN SIE SELBST TRANSPORTMITTEL VERGEBEN? J
MOEGLICHE TRANSPORTMITTEL, (RED.)KAPAZITAETEN, KOSTENSAETZE:
  T  1 T  2 T  3 T  4
  400  200  600  100
    2    3    2    4
KUNDEN, (RED.)BESTELLMENGEN, ENTFERNUNGEN:
  K  1 K  2 K  3 K  4 K  5 K  6
  200  150    0  300  250  400
  800  400  600  500  200  300
VERGEBEN SIE (WEITERE) TRANSPORTMITTEL DURCH ANGABEN UNTER:
TRANSP.MITTELNR.   KUNDENNR.   ZU TRANSP. MENGE
      3                2              50
WEITERE TRANSPORTMITTELVERGABE GEWUENSCHT?   N
EINE KOSTENOPTIMALE TRANSPORTMITTELDISPOSITION - RELATIV ZU
VON IHNEN GETROFFENEN TRANSPORTMITTELVORGABEN - IST:
                         T  1 BELIEFERT
  K  2 K  4
MIT (IN ME):
  100  300
                         T  2 BELIEFERT
  K  5 K  6
MIT (IN ME):
  150   50
                         T  3 BELIEFERT
  K  1 K  2 K  6
MIT (IN ME):
  200   50  350
                         T  4 BELIEFERT
  K  5
MIT (IN ME):
  100
AKZEPTIEREN SIE DIESE TRANSPORTMITTELVERGABE?    J
TRANSPORTMITTELDISPOSITION BEENDET.
```

Lebenslauf

7.3.1947	geboren in Wanfried;
1953 - 1957	Besuch der Volksschule in Kassel;
1957 - 1966	Besuch der Albert-Schweitzer-Schule zu Kassel; Abschluß mit Abitur am 11.2.1966;
1966 - 1972	Mathematikstudium an der Georg-August-Universität zu Göttingen; Abschluß mit Diplom am 11.2.1972;
1972	seit 15.2.1972 wissenschaftlicher Assistent im Fachbereich Wirtschaftswissenschaften der Universität Göttingen am Lehrstuhl für mathematische Verfahrensforschung und Datenverarbeitung.